



Localisation, Tracking, and Navigation Support for Pedestrians in Uninstrumented and Unknown Environments

Carl L. Fischer

Diplôme d'ingénieur, École Supérieure d'Électricité, Gif-sur-Yvette, France.
Master recherche, Université de Rennes 1, Rennes, France.

September 2012

Thesis submitted for the degree of
Doctor of Philosophy.

School of Computing and Communications,
Lancaster University, UK.

Localisation, Tracking, and Navigation support for Pedestrians in Uninstrumented and Unknown Environments

Carl L. Fischer

Thesis submitted for the degree of Doctor of Philosophy

September 2012

Abstract

Research into localisation and tracking of pedestrians is a growing area, but developed techniques have tended to be infrastructure-based or to rely on access to prior information such as floorplans and maps. There are situations, such as search and rescue missions, where more robust and versatile self-contained localisation systems are desirable. To our knowledge, this a relatively unexplored area of research. Relative localisation, involving only objects in the vicinity of the target being tracked, offers a partial solution by not requiring any infrastructure. We demonstrate and discuss some techniques based on this approach, and develop an algorithm suitable for tracking highly mobile sensor networks. We also highlight its limitations and look for complementary solutions. Pedestrian dead reckoning (PDR) based on foot-mounted inertial sensors is a promising method which we describe in detail, including its inherent flaws. We combine the dead reckoning and sensor node techniques to perform simultaneous localisation and mapping (SLAM) for pedestrians in indoor environments. Finally, we include our SLAM algorithm in a complete navigation solution which we evaluate in a virtual environment. This study allows us to offer insight into problems and opportunities offered by these technologies and their application in the field of pedestrian navigation in uninstrumented and unknown environments. We describe the potential offered by tracking and navigation once they are no longer dependent on infrastructure, pre-deployment, or prior knowledge of an area.

Acknowledgements

I am extremely grateful to my supervisor Hans Gellersen for giving me this opportunity, and offering sound advice and encouragement throughout the process. The flexible working hours and constant provision of funding have allowed me to enjoy a life outside of my research, and this has been much appreciated. Many thanks also to Mike Hazas for taking responsibility for my supervision when needed and for helping me see more clearly through some of the lower level details of localisation algorithms.

Throughout my time at Lancaster, the members of the Ubicomp/EIS group, past and present, have made things incredibly enjoyable. Rose, your comments on my first draft have certainly made it a better piece of work, and Ming Ki, I appreciate your last minute review of key parts of this thesis. Thank you all for your friendship and support.

Kavitha and Poorna, it has been a pleasure working with you. Thank you both for your contributions to this thesis, particularly the heated discussions about specific implementation issues. I have also had the privilege to work with many great colleagues on the Relate and Profitex projects. Gerald, thank you for your help with the data evaluation. Markus Klann at Fraunhofer FIT, I am indebted to you for allowing me access to FireSim, and for your inspiring ideas. Mirko and Kathrin, I am grateful for the time you spent talking me through the code. Many thanks to John-Olof Nilsson at KTH for his invaluable insights into parameter tuning for the PDR Kalman filter.

Roger Balderstone, formerly of Lancashire Fire Service, and Nick Clough of Chirange Technologies both gave of their time to discuss with me the needs of British firefighters in terms of navigation support. Their input was insightful and encouraging.

Thank you to my parents and sister for being patient and listening to my complaints. My dear wife Amy, thank you for marrying me even though I still had this to work through, thank you for letting me stay up late at least twice to get it finished, and for your help with qualitative data analysis.

And finally, thank you Lord for all of the above, for making this world such an exciting place, and for giving me the ability to explore, enjoy, and make a difference.

Contents

Acknowledgements	iii
Contents	iv
List of figures	ix
List of tables	xii
List of terms	xiii
1 Introduction	1
1.1 Localisation and tracking: an unsolved problem	2
1.2 Methods	4
1.3 Contribution	6
1.4 Summary of chapters	7

2	Related work	9
2.1	Wireless sensor networks (WSN)	10
2.1.1	Measurements and methods	10
2.1.2	Limitations	12
2.2	Robotics	13
2.2.1	Measurements and methods	14
2.2.2	Limitations	15
2.3	Asset tracking	16
2.3.1	Measurements and methods	16
2.3.2	Limitations	17
2.4	Pedestrian tracking	18
2.4.1	Measurements and methods	18
2.4.2	Limitations	20
2.5	Target tracking	20
2.5.1	Measurements and methods	21
2.5.2	Limitations	21
2.6	Machine vision	21
2.6.1	Measurements and methods	22
2.6.2	Limitations	23
2.7	Localisation for emergency response	23
2.7.1	Requirements analysis	24
2.7.2	Key properties	30
2.7.3	Discussion of systems	32
2.8	Conclusion	36

3	Peer-to-peer ultrasonic measurements for relative positioning	40
3.1	Background and simple demonstrators for relative localisation . . .	41
3.1.1	Hardware: ultrasonic sensor nodes	42
3.1.2	Non-linear regression	45
3.1.3	Using individual measurements	46
3.2	Real-time relative positioning of mobile nodes in sensor networks . .	50
3.2.1	Introduction	51
3.2.2	Related work on localisation in wireless sensor networks . . .	53
3.2.3	Relative Kalman Filtering	56
3.2.4	Real deployments	64
3.2.5	Evaluation	69
3.2.6	Suitability of the relative Kalman filter for wireless sensor network localisation	79
3.3	Discussion on the merits of ultrasound for localisation	84
3.4	Conclusion	88
4	Pedestrian dead reckoning	91
4.1	Introduction	92
4.2	Key references	93
4.2.1	Inertial navigation and Kalman filtering	93
4.2.2	Pedestrian dead-reckoning (PDR) using shoe-mounted iner- tial sensors	94
4.3	Inertial pedestrian dead-reckoning	95
4.3.1	Attitude and heading reference systems	96
4.3.2	Approximations and assumptions	97
4.3.3	Zero-velocity detection	98
4.3.4	Position estimation	101

4.4	Practical implementation	107
4.4.1	Initialisation	107
4.4.2	Zero-velocity detection	108
4.4.3	Main loop	108
4.5	Evaluation	111
4.5.1	Recordings at Lancaster University’s Infolab21	112
4.5.2	DLR figure-of-eight recording	114
4.5.3	Running	114
4.6	Challenges and suggestions	117
4.6.1	Major causes of tracking error	117
4.6.2	Parameter tuning	118
4.6.3	Output format	119
4.6.4	Tracking the orientation	120
4.6.5	Common mistakes	121
4.7	Conclusion	121
5	Simultaneous localisation and mapping using pedestrian dead reck- oning and ultrasonic sensor nodes	124
5.1	Introduction	125
5.1.1	Related work	125
5.1.2	Contribution	131
5.2	Implementation: multi-modal sensing and algorithms	132
5.2.1	Sensing technologies	133
5.2.2	Localisation and mapping algorithms	134
5.3	System evaluation	142
5.3.1	Description of experiments	143
5.3.2	Results and analysis	149
5.3.3	Results summary	162
5.3.4	Improvements and future work	168
5.4	Conclusion	171

6	Design and evaluation of a navigation system using a virtual reality simulator	173
6.1	Introduction	174
6.1.1	Evaluating navigation systems	175
6.1.2	FireSim	176
6.1.3	Contributions	177
6.2	Navigation system evaluation	178
6.2.1	Study overview	178
6.2.2	Navigation system	180
6.2.3	Study procedure	183
6.2.4	Pre-study	187
6.3	Results	189
6.3.1	Pre-study	189
6.3.2	Main study	189
6.4	Discussion	195
6.4.1	Virtual reality as a study tool for sensor-based systems . . .	195
6.4.2	Evaluation of the non-infrastructure-based navigation system	201
6.4.3	Thoughts on using inertial PDR and SLAM	206
6.5	Conclusion	207
7	Conclusion	211
7.1	Contributions	211
7.2	Challenges and lessons learnt	213
7.3	Vision for the future of navigation support for emergency response .	214
A	Inertial pedestrian dead reckoning implementation in Matlab	217
B	Navigation algorithm and implementation issues	223
C	User study: navigation in a virtual environment	227
C.1	Study design	227
C.2	Verbal questionnaire	227
C.2.1	Following the three missions in each building	228
C.2.2	At the end of the study	228
C.3	Demographics questionnaire	229
C.4	Summary of interviews	229
	Bibliography	231

List of figures

2.1	Firefighter training exercise.	27
2.2	Comparison of location systems.	38
3.1	Ultrasonic devices from the Relate project.	43
3.2	Components of the Relate Gateways prototype.	48
3.3	Navigation boots with sensors.	49
3.4	Notations used for Relate measurements.	57
3.5	Outlier filtering.	62
3.6	Ultrasonic sensor node: a Relate <i>brick</i>	65
3.7	Experimental deployment of static and mobile Relate sensor nodes.	66
3.8	Distribution of severe ranging errors and measurement failure rate.	68
3.9	Distribution of measurement rates.	69
3.10	Median accuracy convergence for naive and graph trace initialisation.	71
3.11	Distribution of relative localisation errors for static nodes.	72
3.12	Error distributions in a multihop network.	74
3.13	Multihop initialisation performance.	75
3.14	Kalman filter dynamic tracking accuracy.	77
3.15	Tracking accuracy for different movement models.	78

3.16	Tracking accuracy for different pulse emission rates.	78
3.17	Per-node computational complexity of the relative Kalman filter (floating point multiplications).	80
4.1	Sensor to navigation frame coordinate transformation.	96
4.2	Phases of walking in the PDR algorithm	100
4.3	Naïve zero-velocity update method.	102
4.4	Kalman filter zero-velocity update method.	104
4.5	Kalman filter with orientation estimation.	106
4.6	IMU attached to the instep of the foot.	113
4.7	PDR trace of a four minute walk through the Infolab.	115
4.8	PDR trace of the DLR figure-of-eight data set.	116
4.9	PDR trace for running.	116
5.1	Range and bearing measurement errors.	134
5.2	Notations used for PDR measurements.	136
5.3	Sensor position initialisation.	143
5.4	The test area for the SLAM system.	144
5.5	Experiment and data collection.	146
5.6	Paths estimated from inertial pedestrian dead reckoning alone. . . .	151
5.7	Pedestrian position uncertainty for PDR alone.	152
5.8	Paths estimated from PDR and known sensor positions.	153
5.9	Pedestrian position uncertainty with known landmarks.	154
5.10	Changes in the estimated positions of the sensors.	156
5.11	Final estimated positions of the sensors using SLAM.	158
5.12	Cumulative range error distributions.	160
5.13	Cumulative bearing error distributions.	161

5.14	Cumulative range error distributions for near and far sensors.	163
5.15	Cumulative bearing error distributions for near and far sensors.	164
5.16	Median range errors for each successive section of the path.	165
5.17	Median bearing errors for each successive section of the path.	166
6.1	Structured and unstructured environments used for in the study.	179
6.2	Waypoint guidance along a path.	182
6.3	Changes in navigation interface following pre-study.	184
6.4	Training area for navigation study.	185
6.5	Post-mission questionnaire.	186
6.6	Evaluation of players' performance.	188
6.7	Sample car park traces.	194
6.8	Sample office traces.	196
6.9	Effects of relative navigation.	203

List of tables

2.1	Summary of location support systems for emergency response. . . .	37
4.1	PDR system parameter values.	120
5.1	Parameters used in the SLAM algorithm.	149
C.1	Navigation study design.	228
C.2	User feedback.	230

Introduction

Localisation systems represent a significant area of current scientific and industrial research. Knowing the location of users, robots, sensors or specific objects can enhance existing computer-based applications and opens up new possibilities to assist humans and to automate tasks. There are applications in robotics (Thrun et al., 2000), ubiquitous and mobile computing (Duff et al., 2005), and wireless sensor networks (Niewiadomska-Szynkiewicz et al., 2009), and research into localisation systems covers a range of aspects including data processing algorithms (Giremus and Tourneret, 2006), sensing and measurement hardware (Belloni et al., 2009), and the effects of localisation information on the user experience (Mahmud et al., 2009). Some form of localisation is essential for autonomous vehicles such as military drones or the cars of the DARPA Grand Challenge (Urmson et al., 2008); smartphones typically include several apps which make use of the embedded GPS receiver for localisation; and navigation aids for the visually impaired (Jacquet et al., 2004) rely on sensing their location. Thus, localisation is a key part of modern computer systems.

Localisation facilitates other fields of scientific research and industrial operations: for instance, monitoring the environmental conditions in a field of crops

(Hwang et al., 2010), or the movements (Steiner et al., 2000) and interactions (Mennill et al., 2012) of certain animals, optimising gully cleaning operations¹, or guiding medical tools to perform delicate surgical operations (Bandala and Joyce, 2007), These tasks are made possible, or easier, or more reliable, thanks to developments in localisation systems.

These systems also provide a platform for testing and further developing hardware and algorithms that are useful for other activities. Researchers developing localisation systems rely on communication channels (Stojanovic et al., 2002), networking algorithms (He et al., 2004), and sensor hardware, and are faced with limitations in each of these areas. New developments then overcome these limitations, and the improvements are made available to the wider research community.

1.1 Localisation and tracking: an unsolved problem

Localisation appears to be a solved problem in some contexts, but there are many more where existing solutions do not apply. A number of commercial products already support localisation, and many research projects have already suggested ways of estimating location indoors. GPS is a common solution to outdoor localisation problems. From the end user's point of view, it does not rely on any particular infrastructure or esoteric hardware. However, it does not function well when the receiver is indoors and satellites are out of view, and generally does not provide the resolution required for indoor navigation. Specialised localisation systems pre-installed in a building, or existing infrastructure, such as wifi access points, can provide, or at least contribute, to a solution. But in many places, this infrastructure does not exist at all, is insufficient or may be damaged. For instance, caves, tunnels, or derelict buildings are unlikely to contain even a functional electrical

¹http://www.intouch-ltd.com/gully_cleaning.asp (Accessed 2012.09.24.)

supply, let alone a high-tech localisation system. The same applies to localisation systems which rely on maps and floorplans. The structures where location information would be most useful may well be those for which plans are unavailable or outdated.

Activities such as firefighting, urban search and rescue, or underground exploration present particular challenges when it comes to estimating location. The environment is shielded from radio signals. This makes communication difficult, and localisation based on GPS satellites or other radio transmitters deployed outside becomes unreliable. It also makes it difficult to deploy any devices directly into the area of interest (e.g., dropped from a plane) without first entering and exploring it. In most cases, there will not be any specialised localisation system, and any existing infrastructure will be unreliable due to the conditions. Maps and plans can be a tactical asset when they are available, but they can also be misleading as they cannot indicate passages which are blocked by a collapsed roof, walls which have been broken through, or areas which have never been mapped.

We focus our research on localisation in environments with the following characteristics:

Uninstrumented: No existing localisation or communication system. This includes indoor environments with no GPS or communication with outside, and restricted access.

Unknown: No floorplans, maps, or other prior knowledge. No assumptions about the existence or geometry of corridors, stairs, or doorways (unstructured).

The terms localisation, tracking and navigation appear frequently in the literature, and are often used interchangeably. In our work, the distinction is important due to the lack of absolute frame of reference, so we define the terms here. *Localisation* refers to determining the position of a person or object at a particular

instant. When we estimate its position over time by logging movement (rather than positions) (Fischer et al., 2013), we prefer the term *tracking*. We can also track a target by successively performing localisation at several points, or by modeling its trajectory (Park et al., 2008). *Navigation* is providing the user with directions to help them achieve their goal, which could be reaching a particular location, following a path, or avoiding obstacles. A navigation system could use localisation and tracking information along with a map to provide guidance. But a person could also follow directions from one waypoint to the next in the absence of a complete map. In many cases, we will take the term *position* to include both the spatial coordinates and the orientation. The *orientation* relative to the environment is essential when providing navigation support as the direction in which we guide the user depends on it. Throughout this thesis, we emphasise the distinction between *relative* and *absolute* localisation. The exact difference is context dependent, and it could be argued that all localisation is relative. However, for the purposes of our work, we define relative localisation as the estimation of the coordinates of a target object relative to other objects within the close vicinity of the target, using only measurements to and from these objects. Absolute localisation, in contrast, is the estimation of coordinates within a frame of reference that extends far beyond the immediate vicinity of the target, and using measurements from far away objects.

1.2 Methods

We ask which technologies, methods and algorithms can be used to provide localisation, tracking, and navigation support in unknown, uninstrumented, and unstructured indoor environments.

The localisation problem has not been solved for such environments, but we acknowledge that it has been approached from many angles by researchers from different areas, often with a particular application in mind and a specific set of

constraints, such as the availability of sensors, the type of movement, or the required accuracy. In this thesis, we provide some insight into how existing research can be used to address the challenges we have described, which aspects are suitable, which need further developments in terms of algorithms or sensing technology, and which combinations result in a robust system.

Due to the broad range of areas which relate to our topic, in our survey of existing work we report on general methods and approaches, rather than small variations in performance between different versions of an algorithm for instance. We do, however, perform small focused studies into particular technologies and algorithms. When we recognise a clear limitation of one method, we attempt to address it by drawing from a different area. We evaluate some of the algorithms using controlled experiments, but the simpler concepts are directly tested with a demonstration system. In cases where the technology currently available to us is limited or inadequate, we ask users to test our ideas in a virtual environment. This is an important aspect of our work because, although it does not permit a detailed evaluation of an algorithm or specific type of sensor, it allows us to see how a localisation system could be deployed and used in practice, and whether it is convenient or reliable from the user's perspective. We were not able to conduct field trials in the course of this work due to the lack of suitable sensor platform and the difficulties of safely creating a low visibility environment.

A search and rescue mission, performed by firefighters for instance, is a canonical example of a scenario in which a robust tracking and navigation system would be valuable. This is simple to understand even with very little background knowledge of firefighting. During such a time critical mission, the rescuers do not have the opportunity or time to deploy and configure a localisation system, and the buildings in which they operate are often damaged. This application has driven much of the research in this thesis and has helped us understand more clearly the shortcomings of existing work. In the following chapters, we refer back to this example scenario to evaluate the suitability of the different methods under investigation.

1.3 Contribution

Despite recent technical developments in indoor positioning and pedestrian navigation, and a number of research projects which aim to support emergency response through the use of computers, there is very little in the literature about truly self-contained navigation systems. Few researchers, if any, have explicitly identified, let alone addressed, the challenge of localisation in unknown and uninstrumented environments. In this thesis, we contribute the following:

- We define localisation in unknown and uninstrumented environments as a specific and relevant challenge, worthy of investigation.
- We show that existing research in a range of areas offers some useful techniques but that none of these constitute a complete or general solution.
- We give an overview of research into localisation for emergency response and identify unanswered research questions.
- We develop and test several combinations of technologies and algorithms which address some aspects of our problem:
 - a Kalman filter for the relative localisation of mobile sensor nodes using ultrasonic distance and bearing measurements,
 - a pedestrian dead reckoning algorithm based on a foot-mounted inertial measurement unit,
 - a simultaneous localisation and mapping algorithm built on the above, and suitable for pedestrian navigation.
- We highlight the challenges of testing navigation systems (as opposed to localisation systems) and we show how these can be resolved through virtual reality simulations.

1.4 Summary of chapters

In each of our chapters we examine different aspects of solving this challenging localisation problem, and draw conclusions about the effectiveness of methods taken from wireless sensor networks, robotics, and mobile and ubiquitous computing.

In chapter 2 we give an overview of work in these areas and highlight the limitations that prevent it from solving our localisation problem. We also point out to what extent certain features of sensing technologies and localisation methods contribute to our research. One section looks more closely at the special case of localisation for emergency response and the state of the art in commercial and research projects in that area in order to understand why this problem is still so challenging.

In chapter 3, we take a look at ultrasonic range and bearing sensing, a technology used in the area of wireless sensor network localisation. We use these sensors in the design and trial of a wearable navigation system suitable for deploying during a search and rescue mission. We then address the practical failings of this system by bringing in a more advanced algorithm to estimate locations more robustly but without losing the features which make the initial concept attractive; we test this algorithm in a controlled wireless sensor network.

In chapter 4, we cover inertial pedestrian dead-reckoning (PDR), a technique which has been used by many projects in recent years to build pedestrian tracking systems, yet is rarely described in detail. We give a practical description of how PDR works, along with a detailed implementation. Despite its attractive features for our research, PDR is not of itself a complete solution for all our tracking needs.

Drawing on robotics research, in chapter 5 we show how PDR can be combined with the ultrasonic beacon concept proposed in chapter 3 to bring together the best

of both methods and to address each of their limitations in a single localisation and tracking system which could offer a solution to navigation for firefighters.

In chapter 6, we bridge the gap between localisation and navigation by implementing this system in a video game and seeing how successful players are at using it to find their way. This allows us to study how human beings, who already have their own sense of direction and individual navigation strategies, respond to the information computed by our system, and how it copes when users ignore its indications or break assumptions that we have made.

Finally, chapter 7 concludes this thesis by summing up our contributions to the field of localisation and identifying which areas need further work.

Related work

Several areas of research investigate the issue of localisation and offer solutions to estimate the position of devices, objects or people. This is a very broad topic which is studied for a variety of reasons. Sometimes it is a key part of a novel application for mobile phones (Siegemund and Flöer, 2003; Guinard, 2007), or a component of a defence system (Groves, 2003; Chen et al., 2009; Simon et al., 2004), or a means to illustrate (Simon, 2000) or validate (van der Merwe, 2004, ch.5) the theory behind certain families of algorithms. In this chapter, we look at six areas which contain valuable tools and methods for our work: wireless sensor networks, robotics, asset tracking, pedestrian tracking, (military) target tracking, and machine vision. We characterise their general approach to localisation, in terms of the technologies and physical properties used to obtain low level measurements, and in terms of the mathematical and geometrical properties which connect these to the actual location estimates. The wide range of techniques and the explorative nature of this research make it difficult, and even unnecessary, to perform a detailed quantitative comparison. At this stage, it is most important to determine which general categories of technologies and algorithms are potentially suitable. Therefore, we draw a more qualitative overview of the higher level characteristics of these systems. In particular, we highlight the reasons why none of them present a complete solution

to our localisation problem. We close this chapter with a case study of localisation for emergency response in which we look at the requirements and challenges, and see how some of these principles have already been applied to emergency response scenarios, either in research projects or commercially.

2.1 Wireless sensor networks (WSN)

Work in the area of wireless sensor networks (WSN) is typically motivated by the idea that many small, wireless, electronic devices can monitor the physical properties of our world more effectively than a single monolithic system. Applications include monitoring temperature and humidity in a field of crops (Hwang et al., 2010), analysing the performance of a factory production line (Valverde et al., 2012), or detecting threats in a battlefield (Dargie and Poellabauer, 2010). WSN researchers may consider algorithms for data aggregation and pattern recognition, routing protocols for fast and reliable communication of data, and localisation methods so nodes can tag data with the location at which it was recorded. Since the sensor nodes are often assumed to be battery-powered and deployed in harsh environments, researchers generally design and evaluate these algorithms with energy efficiency and resilience in mind (Xia et al., 2011; He et al., 2004). Because of the low computational power available on wireless sensor nodes, many of the developments in localisation algorithms are optimisations designed not only to improve accuracy but also to reduce their complexity and the required hardware.

2.1.1 Measurements and methods

Typically, wireless sensor nodes communicate via radio, light, or sound. Radio signals decrease in amplitude with distance. Thus, by measuring the amplitude of the radio signals transmitted by a sensor node, we can estimate its distance using

only the hardware that we already have (Whitehouse et al., 2007). By collecting distance estimates between many pairs of nodes, we can estimate their relative positions using optimisation algorithms. Gradient descent is one such algorithm which attempts to minimise the difference between the measured ranges and the ranges which result from the estimated positions (Hazas et al., 2005). The mass-spring metaphor is another way to iteratively estimate positions by identifying each distance measurement with a spring of a certain length, and each node with a mass subject to the forces of the springs (Priyantha et al., 2003; Efrat et al., 2006). By applying the laws of classical mechanics, each mass relaxes into a position which is close to the true position of the corresponding node.

Alternatively, we can ignore all distance estimates and only consider connectivity. We assume that if nodes are able to communicate, they are within a limited range of each other. This constrains their positions in space and allows us to estimate their positions using various geometric algorithms (Baggio and Langendoen, 2008; Xiao et al., 2008; Stoleru et al., 2007).

There are other methods of measuring distance. Ultrasound, acoustic (audible) sound, and light (Krohn et al., 2005) also decrease in amplitude with distance, and we can use these types of signals to estimate distance between devices. However, for all these methods, the relationship between amplitude and distance is very dependent on the environment and often unpredictable.

Instead of the amplitude, we can measure the time of flight of radio or sound waves (Guo and Hazas, 2011). Although the speed of sound can vary with temperature and humidity, the results tend to be more accurate and robust than using amplitude. We can measure the time of flight of sound waves directly by using a radio signal to synchronise the nodes, because radio waves propagate so much faster than sound. We can also measure round-trip time of flight by sending any type of signal and measuring the time until we receive a reply (Thorbjørnsen et al., 2010). Or we can use the time difference of arrival in the case where one

node transmits and several others listen (Muthukrishnan and Hazas, 2009). The difference in arrival time gives us a relationship between the positions of the transmitting and receiving nodes. This method is also suitable for locating acoustic sources (Guo and Hazas, 2011) and ultrasonic sources (Duff et al., 2005) when the locations of the other sensors are known. Angles of arrival, measured using arrays of radio antennae (Belloni et al., 2009), or microphones (Ali et al., 2007), or ultrasonic transducers (Hazas et al., 2005; Priyantha et al., 2001), are more complex to acquire but are also suitable for position estimation using similar optimisation algorithms (Peng and Sichitiu, 2006). A more comprehensive overview of localisation in sensor networks is given by Pal (2010).

2.1.2 Limitations

Many wireless sensor network localisation algorithms assume there will be a dense and reasonably uniform mesh of sensor nodes measuring the physical properties of the environment and communicating these values to each other or to a base station. These algorithms will therefore often be less reliable or completely unsuitable when the network is sparse, with fewer nodes spread over a greater area, and when the nodes are deployed in a non-uniform configuration, such as a line. The focus of WSN research has changed to include tracking of fast moving objects for instance, but many of the techniques and algorithms still rely on the idea that we are interested in environmental properties which change slowly, and that the sensor nodes themselves are not moving. Therefore, many algorithms are not designed for highly dynamic situations and the update rates are not sufficient to deal with fast moving sensor nodes.

WSN localisation algorithms also take advantage of the large number of nodes to average out measurement errors, and many that are claimed to be suitable for “sparse” networks require a high degree of connectivity and a geometry that are not compatible with deployment on-the-fly by pedestrians (e.g., Wang et al.

(2008), Goldenberg et al. (2006)). Some do not deal well with errors that are not white noise. In particular, measurements that have systematic non-zero error due to unpredictable environmental factors can cause large errors in the position estimates (Krohn et al., 2007).

A major problem with many of the algorithms described above is that they rely on a reasonable initial estimate of node locations which is then refined. When we use only distance measurements, there is an inherent ambiguity because we have no notion of orientation. But even when angular measurements are available, the optimisation algorithms can fall into local minima and fail to converge to an accurate solution.

Sometimes we can assume that a small proportion of nodes know their absolute positions, thanks to an on-board GPS receiver or a manual calibration phase. This helps with the initialisation and avoids some geometric ambiguities because these nodes are effectively already initialised to their true positions. However, this requirement makes the algorithms less versatile.

2.2 Robotics

Robotics is often concerned with creating machines or vehicles that are able to autonomously navigate in their surroundings. This includes unmanned aerial vehicles (UAVs or drones) (Kim and Sukkarieh, 2007), autonomous underwater vehicles (AUVs) (Olson et al., 2006), and “cars that drive themselves” (Urmson et al., 2008), as well as specialised industrial or military vehicles and experimental humanoid robots (Stasse et al., 2006). In order to achieve this, they need to have a sense of their environment and their position in it. The area of robotics research concerned with this is called simultaneous localisation and mapping (SLAM). This name highlights the fact that building a map when the location of the robot is known, and

locating a robot in a known environment, are relatively simple tasks that can be achieved by recording and matching a series of photos for instance, but that doing both simultaneously is more challenging.

2.2.1 Measurements and methods

In order to navigate autonomously, a robot needs to avoid obstacles, such as walls and furniture for indoor robots, or rocks and trees for outdoor ones. Laser range finders (Konolige, 2004; Hähnel et al., 2003) and sonar arrays (Kleeman, 2003) are popular sensors for autonomous robots because they give a 360 degree view of obstacles around the robot, including their distance. This gives us content to construct a map, but we also need to know how much the robot moved between each set of measurements (or between each scan).

There are several ways to estimate the movements of the robot. If we are controlling it with our own software, then we know how much we are asking it to move at any moment. The true movements may not match exactly what we requested but they give us a rough estimate. If the robot has wheels or tracks, we can perform odometry by measuring how much each wheel has turned and by applying geometric principles to estimate the total movement. This is also subject to measurement error, especially if the wheels slip (Kleeman, 2003). Inertial measurements are also a way of measuring movement. Accelerometers measure changes in speed (accelerations) and gyroscopes measure changes in orientation (rate of turn). By integrating these values, we can estimate how much the robot has moved or turned (Walchko et al., 2003). Or, if the successive laser scans are close enough together, we can perform odometry without any additional sensors, by aligning the scans and analysing the differences between them (Hähnel et al., 2003). We can use a camera in the same way (Tardif et al., 2008).

The robot can use other sensors to build a map of the environment without specifically detecting obstacles. RFID tags, wifi access points, and visual markers

such as QR codes can be detected with the appropriate hardware and used as landmarks in the map.

There are a range of approaches to SLAM. They are mostly probabilistic methods, but they vary by the way they represent uncertainty in the map and the position of the robot, and how they aggregate the measurements (Durrant-Whyte and Bailey, 2006; Bailey and Durrant-Whyte, 2006). It is essential to represent the uncertainty in the estimated values and at least some of the correlations between them in order to get a working system. The positions of the robot and the landmarks can be represented as clusters of particles, or as Gaussian variables with a mean and a standard deviation, or a combination of the above. There are tradeoffs with each representation, especially between robustness and stability over time, and memory and computational requirements. Some algorithms which have been shown to work in practice for short experiments, have been proven to eventually degenerate (Bailey et al., 2006).

2.2.2 Limitations

The principal difference between tracking humans and robots is the type of motion. Robots and vehicles often have more predictable behaviours than humans because they are controlled by software of limited complexity and only have a limited number of actuators. Wheeled or tracked robots generally move horizontally, thus ensuring that laser scans are all taken in the same plane and can easily be matched. They also generally consist of rigid bodies, so all sensors remain in the same positions relative to each other. The mechanics of a robot permit more approximations than human motion, thus making the tracking problem simpler in many respects. A few researchers have considered the more challenging problem of tracking all-terrain robots. Kleiner and Dornhege (2007) essentially use a vision-based system combined with an inertial measurement unit (see the following sections), while Suthakorn et al. (2009) use a laser range finder and accelerometer

but do not detail their method or results. Despite these differences between human and robot motion, the algorithms used in robotics demonstrate principles which can be transposed to pedestrian tracking.

2.3 Asset tracking

Many types of businesses want to keep track of their assets in order to cut costs by optimising their processes and preventing theft. In some cases, it may be practical to track a fleet of vehicles by transmitting GPS coordinates over the GSM network. But for indoor environments, the methods used range from hand-labelling computing equipment, tools, and other valuable assets, to using machine readable tags that have to be manually scanned, to installing highly specialised tracking equipment throughout the premises. We are interested in the latter.

2.3.1 Measurements and methods

The key to many of these asset tracking systems is reliable remote detection. We can use RFID tags if the readers are placed strategically along a production line or in doorways, bottlenecks where items are forced to pass. Caterpillar use such a system for tracking parts (O'Connor, 2007), and Savant provide the EuroSDS tracing system for decontamination of surgical instruments¹; both these systems use RFID tags. NFC tags have similar uses but the short detection range requires that they be manually swiped. For instance, ENAiKOON provide inventory and personnel tracking solutions based on NFC². Systems based on ultrasound signals benefit from the fact that ultrasound is completely blocked by walls and doors³. An ultrasonic pulse transmitted by a special tag will usually only be detected

¹<http://www.savant.co.uk/product/eurosdgs> (Accessed 2012.09.24.)

²<http://www.nfc-tracker.com/> (Accessed 2012.12.28.)

³<http://sonitor.com> (Accessed 2012.09.24.)

by receivers in the same room. This makes room-scale location straightforward. Ultrawide band radio (UWB) systems using time difference of arrival (TDOA) and angle of arrival (AOA) need significantly more receivers and extensive calibration but give higher resolution tracking⁴.

Since many buildings are now completely covered by wifi, we can use the wireless access points to estimate the location of a networked device⁵. Each access point covers a limited area, so at any given time, the visible access points and their respective signal strengths give us a rough idea of our location. These fingerprints can be recorded at different locations during a calibration phase and stored in a database as a reference. Or if the locations of the access points are known, we can perform some basic interpolation to determine our approximate location (Mok and Retscher, 2007). Wifi fingerprinting is appealing because of the many access points that exist, especially in urban areas and in office buildings, and the fact that most laptops and smartphones are already equipped with wifi cards.

2.3.2 Limitations

For many asset tracking applications, high update rates and fine resolution are unnecessary. The goal is to know roughly where the company's equipment is, or where it was last used, rather than locating it in real time. Many of the items will remain in the same position for a reasonably long time, giving ample opportunity for them to be detected. This is not suitable for continuous tracking of highly dynamic objects. In addition, Elnahrawy et al. (2004) state that there are fundamental limits to the accuracy of localisation systems based on signal strength (median localisation error of 10 feet using commodity 802.11 hardware).

⁴<http://www.ubisense.net/en/rtls-solutions/research-packages.html> (Accessed 2012.09.24.)

⁵<http://www.ekahau.com> (Accessed 2012.09.24.)

These systems are built around electronic hardware pre-attached to the building at specific points, and they rely on infrastructure for power and communication. For these reasons, they are limited to controlled and specialised environments.

2.4 Pedestrian tracking

Automatic tracking of people is a relatively new area which has become more relevant and achievable as electronic devices have become smaller and the majority of people in our societies now carry at least a mobile phone with them. Many recent phones include a GPS receiver, and can also benefit from services such as Skyhook⁶ which additionally uses cell tower and wifi fingerprinting. Developers of social networking applications are trying to make use of this location-awareness (e.g., foursquare⁷, and Facebook’s “check-in” service⁸). However, one of its main applications remains GPS-assisted navigation.

Indoor pedestrian tracking is still an open problem that is relevant for the development of ubiquitous and embedded computing systems such as smart-homes (Mrazovac et al., 2011) and smart-hospitals (Coronato and Esposito, 2008). Many of the solutions described previously are applicable in such scenarios, but in this section, we discuss more novel methods of tracking specific to pedestrians.

2.4.1 Measurements and methods

Pedestrian motion has certain features that we can use for tracking. We have two ways of performing dead-reckoning that are specific to pedestrians. In the first method, we count steps and estimate the direction of travel (Randell et al., 2003). By accumulating each step and its direction, we get an idea of the trajectory of

⁶<http://www.skyhookwireless.com/> (Accessed 2012.09.24.)

⁷<https://foursquare.com> (Accessed 2012.09.24.)

⁸<https://www.facebook.com/about/location/> (Accessed 2012.09.24.)

the pedestrian. Steps are assumed to have a constant length, or their length can be estimated based on factors such as the delay between steps or the maximum and minimum accelerations. We can conveniently implement such a system on a mobile phone worn in a pocket or held in the hand, using the accelerometers and compass that already exist in the phone. The alternative is to use a separate inertial measurement unit (IMU) and double integrate accelerations and gyroscope readings to obtain an estimate of the trajectory (Foxlin, 2005). In principle, this should work for any type of motion, but due to the characteristics of the sensors, the results are unusable as error accumulates very fast. By attaching the inertial sensors to the foot, we can reset the estimated velocity to zero every time we detect a footfall, thus reducing the error to acceptable levels. This is a very effective “trick”, but one that only works for pedestrian motion with foot-mounted sensors. Both these methods are completely self-contained and do not require any external hardware.

Pedestrian dead-reckoning (PDR) inherently accumulates error and provides no means to correct absolute position estimates. For this reason, many researchers try to combine it with other location information in order to make the system more robust. GPS provides absolute position estimates when outdoors, and dead-reckoning can continue to provide position estimates for a few minutes when the pedestrian moves indoors and GPS becomes unavailable (Godha and Lachapelle, 2008). Wifi fingerprinting described earlier can be used in a similar way (Evennou and Marx, 2006). In some situations, it may be acceptable for the user to manually correct their position from time to time by clicking on a map or floorplan.

Where maps or floorplans are available, we can also use the information they provide to improve location estimates (Krach and Robertson, 2008; Beauregard et al., 2008). By assuming that the tracked pedestrian only moves along corridors and through doorways (without ever passing through a wall), we can avoid some of the error that creeps into our estimates. Or using a related technique, we can

try to recognise, in the inertial data, certain patterns of movement corresponding to stairs or ramps, or perhaps use an additional sensor to recognise doorways, and then correct the estimated position using the corresponding positions on the floorplan (Gusenbauer et al., 2010; Jiménez et al., 2011). This will often require us to use a probabilistic representation of the position because there is ambiguity between several similar features at different locations on the map.

2.4.2 Limitations

The main drawback of any dead-reckoning technique is the inherent drift. The error in position estimates gradually increases unless we use an independent source to determine the absolute position and correct the estimate. This means we lose the benefits of completely self-contained PDR and start relying on external infrastructure or detailed prior knowledge of the environment. Improvements in MEMS inertial sensing may mean that in a few years we will be able to track a pedestrian for several hours with almost no error, but even then, a single glitch in measurement values or timestamps will compromise all future estimates.

We achieve the most reliable tracking by using dedicated foot-mounted sensors which require additional wiring and specific attachments. Even the most recent MEMS inertial sensors do not allow us to track with the same accuracy if they are mounted elsewhere on the body, or embedded in a handheld device. This presents a practical obstacle to using PDR in many real world applications.

2.5 Target tracking

Tracking aircraft is a key part of most national defence systems and as such has benefited from years of applied research. Using data from radar or cameras, we can not only track, but also predict the position of a fast moving target, even

when it is occluded or when the measurements are noisy. Many implementations of the Kalman filter have been developed specifically for this purpose (Singer, 1970; Gutman and Velger, 1988; Bilik and Tabrikian, 2006) and the maturity of research in this area suggests that it is worth mentioning.

2.5.1 Measurements and methods

A key feature of target tracking literature is the variety of movement models used for the target (Li and Jilkov, 2003). The Kalman filter (or other tracking algorithms) use these models to filter outlier measurements, smooth the estimated trajectory and predict future positions. Different models make different assumptions about how the target moves, how fast it can turn, accelerate, decelerate, what types of manoeuvre are feasible and how likely they are.

2.5.2 Limitations

The range of movements that a ship or aircraft can perform are more limited than those of a pedestrian and thus easier to predict. In many situations, pedestrians may walk in straight lines at constant speed, and turn at right angles because they are following the layout of a building, but not all buildings are built on a grid pattern and these assumptions do not help much in open spaces. More importantly, the situations where tracking pedestrians is critical are those situations where the movement is least normal and least predictable. Thus, methods which work for target tracking are not directly applicable to pedestrians but could potentially be adapted.

2.6 Machine vision

Perhaps machine vision comes closest to mimicking the way humans usually locate themselves and navigate their environment. Similar to our eyes, cameras can give

us a lot of high resolution and high frame rate information about our surroundings. Typically, as we walk around, we identify key features to use as landmarks in a mental map of our environment and to help us locate ourselves (Lee and Tversky, 2005). We later use those same landmarks to determine where we are, whether it's over a few seconds while we are distracted by something else, or much longer term when we are trying to find our way back to a friend's home.

2.6.1 Measurements and methods

We can use machine vision to perform simultaneous localisation and mapping (SLAM), as described earlier. As in other types of SLAM, the defining characteristics are the type of landmark used, and the representation chosen for the state estimates, their uncertainties and correlations (e.g., Gaussian, mixture of Gaussians, cloud of particles). Artificial markers, such as black and white printed patterns similar to bar codes are one type of landmark that are relatively easy to locate in a video stream. They have the added benefit of being uniquely identifiable, making the SLAM algorithm simpler. But natural features such as corners and edges of doorways or other building fixtures are a viable alternative (Shi and Tomasi, 1994; Lowe, 2004). We can store a small image patch for each of the features we choose in order to recognise them later, or encode each feature as a series of numerical values computed from this image patch using feature descriptors such as SIFT (Lowe, 2004) or SURF (Bay et al., 2008).

Visual SLAM can rely either on a stereo camera rig (Davison, 1998) which provides additional depth information, or a single camera (Davison et al., 2007). Researchers have shown it is possible to create a map of a large outdoor environment by simply walking through it with a single camera (Clemente et al., 2007). Others have created a detailed three-dimensional model of a desktop by slowly panning the camera across it (Newcombe and Davison, 2010). More recently, re-

searchers have used the depth-camera from the Microsoft Kinect to map a complete room with a lot of detail (Izadi et al., 2011).

2.6.2 Limitations

The fundamental limitation of visual SLAM is its assumption that the environment does not change. This is not true in all situations. Most SLAM algorithms will filter out a small number of dynamic objects in the scene, such as pedestrians or vehicles passing by. Some are even designed to take advantage of moving objects in the environment (Bibby and Reid, 2007). But as far as we know, none of them are designed to cope with drastic changes in the appearance of the environment itself. During firefighting missions, the surroundings are likely to change substantially due to collapsing floors and ceilings, or walls blackened by smoke. Changing lighting conditions, or low visibility also compromise vision-based techniques.

Scalability is another concern. With most SLAM algorithms, the time required to process each new image increases with the number of landmarks in the map. Due to the resolution of the images and density of the map, the map can only reach a limited size before the processing is slower than real-time. Some algorithms address this by creating a hierarchy of local submaps of limited size, and only working on one local map at a time (Blanco et al., 2008b).

2.7 Localisation for emergency response⁹

Although localisation is becoming available for the general public and for businesses via widespread use of GPS receivers and commercial indoor location systems¹⁰ many solutions are not suitable for use by emergency responders such as

⁹This section is a revised version of: Carl Fischer and Hans Gellersen. Location and Navigation Support for Emergency Responders: A Survey. *IEEE Pervasive Computing*, 9(1):38–47, January–March 2010.

¹⁰<http://www.ubisense.net>, <http://sonitor.com>, <http://www.ekahau.com> (Accessed 2012.09.24.)

firefighters, as highlighted in previous sections. The conditions they work in are significantly more demanding than non-emergency environments. Darkness, smoke, fire, power cuts, water and noise can all prevent a location system from working, and heavy protective clothing, gloves and facemasks make using a standard mobile computer impossible. In the past decade much research effort has been put into this challenging problem and a wide variety of ideas have been developed. Previous surveys have addressed localisation methods in general and have not taken into account the specific requirements of emergency response, e.g., Hightower and Borriello (2001). In this section, we look at different localisation technologies and techniques that could assist responders in the challenging conditions they face.

2.7.1 Requirements analysis

Location and navigation support is useful in many every day situations but essential in emergency response scenarios. Teams need to be able to reach safety quickly if conditions become too dangerous, and the incident commander needs to keep track of where teams are. The simple task of finding one's way in a building becomes a challenge when there is little or no visibility due to smoke and darkness. The high levels of mental and physical stress add to the difficulty. Getting lost in a burning or collapsing building can have fatal consequences for both the rescue personnel and the casualties as breathing apparatus run out of air and medical attention is delayed.

Concrete problems

A report by the National Fire Protection Association (NFPA) (Fahy, 2002) in the United States identifies “lost inside” as a major cause of traumatic injuries to firefighters. Reports¹¹ by the National Institute for Occupational Safety and

¹¹Fatality Assessment and Control Evaluation (FACE) Program, and Fire Fighter Fatality Investigation and Prevention Program (FFFIPP): <http://www.cdc.gov/niosh/fire/> (Accessed 2012.09.24.)

Health (NIOSH) also reveal that disorientation and failure to locate victims are contributing factors to firefighter deaths, and a report on the Worcester warehouse fire, in which six firefighters died, highlights the difficulty to keep track of firefighters within the building as one of the major causes for loss of life (Anderson, 1999).

In case of a sudden increase in temperature a firefighter may only have seconds to reach safety. They need to find the exit as fast as possible. In some cases they may not be able to retreat along the same path due to a collapsed ceiling or floor. Alternative exits may be available but not clearly visible. When a firefighter radios a distress call because they are trapped, or when someone fails to report, the rescue team must be able to locate them. Even when situations are not immediately life-threatening, precious time can be wasted by searching the same room twice or failing to search another. The incident commander also needs to know elements of the building layout, where the team members are and which parts of the building have been searched.

Several recurring recommendations from the NIOSH reports explicitly highlight the need for a navigation and tracking system, and suggest some solutions:

- “train fire fighters on actions to take if they become trapped or disoriented inside a burning structure” (FACE report 2007-18¹²);
- “consider using exit locators such as high intensity floodlights, flashing strobe lights, hose markings, or safety ropes to guide lost or disoriented fire fighters to the exit” (FACE report 2007-18¹²);
- “ensure that the Incident Commander receives pertinent information (i.e., location of stairs, number of occupants in the structure, etc.) from occupants on scene and information is relayed to crews during size-up” (FACE report 2006-19¹³);

¹²<http://www.cdc.gov/niosh/fire/pdfs/face200718.pdf>

¹³<http://www.cdc.gov/niosh/fire/pdfs/face200619.pdf>

- “working in large structures (high rise buildings, warehouses, and supermarkets) requires that fire fighters be cognizant of the distance traveled and the time required to reach the point of suppression activity from the point of entry” (FACE report 2007-18¹²);
- “conduct research into refining existing and developing new technology to track the movement of fire fighters inside structures” (FACE reports 2007-18¹² and 2008-09¹⁴).

In addition to the localisation and navigation requirements, other reports emphasise the need for reliable communication of interior conditions to the incident commander and for monitoring building stability (FACE reports 2007-16¹⁵ and 2007-01¹⁶). Temperature, smoke, sounds and vibrations are all indicators of the progression of the fire and the stability of the building.

Current practices

Firefighters have developed their own specific navigation practices for use in poor visibility. Details vary but overall the same ideas are used worldwide. The methods tend to be simple and practical, and the equipment is seemingly low-tech and very robust.

Techniques Following a hose is a simple method to find the exit through a dark or smoke-filled building. If no hose is available, firefighters may use dedicated ropes called *lifelines* which connect them to a point outside the dangerous area (Fig. 2.1). The other end can be left attached if a new team comes in to continue the search (Klann, 2009). Additional lines may be attached to rings on the main lifeline to allow several firefighters to branch off in different directions while remaining

¹⁴<http://www.cdc.gov/niosh/fire/pdfs/face200809.pdf>

¹⁵<http://www.cdc.gov/niosh/fire/pdfs/face200716.pdf>

¹⁶<http://www.cdc.gov/niosh/fire/pdfs/face200701.pdf>

physically linked to the rest of their team. A series of knots on the main lifeline helps firefighters determine the direction and distance to the exit and can be used as reference points when radioing positions to the commander (Sendelbach, 2002).

A flashlight left in the doorway of a room helps locate the exit and indicates to colleagues that the room is currently being searched, and a chalk mark on the door indicates that a room has already been searched (Sendelbach, 2002; Klann, 2009). Teams returning from a search mission sketch the layout of the building to assist the commander and any further teams.



Figure 2.1: Two Paris firefighters practise using a lifeline with their facemasks blacked out. *Photo courtesy of Markus Klann, Fraunhofer FIT.*

All firefighters entering hazardous areas wear a Personal Alert Safety System (PASS) device attached to their breathing apparatus (National Fire Protection Association, 2002) (as cited by Donnelly et al. (2006)). The PASS device sounds an alarm if the firefighter does not move for a short time. At a fire scene, the sound of a PASS alarm is a signal that a firefighter is in distress. By following the sound the rescue team can locate that firefighter. While not strictly a navigation tool, thermal imaging cameras can also be used for finding people and seeing walls, doorways and windows when unaided vision is obscured by smoke.

Many firefighters are trained to search a dark room while keeping either their left or right hand in contact with the wall. This helps with orientation and provides a strategy for systematically exploring an unknown space (International Association of Fire Chiefs, 2004).

Human contact and accountability are also essential. Searches are always performed in teams of at least two members who should avoid being separated (Clark, 1991). During a lifeline search, one team member may remain at a fixed position to help with orientation and provide progress reports while their colleagues search further. Locations are reported as accurately as possible over the radio to the commander outside the building who keeps track of team locations on a whiteboard (Jiang et al., 2004).

Limitations of traditional methods These methods are practical and simple to understand, and they become more effective with training. However, they sometimes fail. A lifeline may become tangled in furniture, a flashlight may be buried under debris, and the temperature of the environment may make a thermal imaging camera unusable. But the principles behind these methods are familiar — the physical properties of a rope, the propagation of light, even the principle of thermal imaging. Failure is understood and even expected in certain conditions. The left hand method for finding an exit can also be misleading and a person can find themselves walking in circles around a large pillar or repeatedly visiting two or three rooms connected by several doors.

None of these techniques are treated as fixed ways of operating. They are used to aid and support navigation rather than impose an inflexible method. Human error can occur especially during complex and prolonged incidents. Simple techniques such as taking notes (for the commander) or following a rope (for the search teams) are designed to reduce the mental load. As pointed out in the NIOSH reports, many improvements can be made by following procedures and through adequate training.

But localisation, sensing and communication are all areas where embedded computers, body worn sensors, and wireless sensor nodes could play a role if they can be adapted to the harsh conditions and accepted by highly trained professionals. High-tech systems have potential not only to address the limitations of traditional methods, but also for adding value beyond what is currently possible.

Constraints on high-tech location systems

Navigation by sight is impossible when darkness, smoke, or dust limit visibility to less than an arm's length. Persons or objects that are out of reach can easily be passed unnoticed. The environment can change as ceilings, floors, or shelves collapse, as furniture is moved, and doors are opened or closed by people searching for an exit. The noise of the fire can mask PASS alarms, interfere with radio conversations and make cries for help difficult to locate.

High-tech systems are generally not adapted to these conditions. Propagation of radio, ultrasound and laser signals typically used for location is hindered by high temperatures, thick smoke, noise, gusts of air, obstacles and falling debris. A report by the City of Phoenix Fire Department (Worrell and MacFarlane, 2004) analyses problems with radio communications inside buildings and identifies unreliable radio links as the cause of several injuries. Sensors deployed in the environment may be kicked, fall through the floor, or be buried. Firefighters may crawl or walk in unusual patterns, and body-worn sensors may lie at odd angles. In addition, there is the issue of presenting the right amount of information to the firefighter in an accessible way, and ensuring that devices can be used in the dark with gloves. Finally, the casing and electronics of all devices must be made as robust as possible, in the same way as PASS devices and radios, to withstand rough handling and very high temperatures (Donnelly et al., 2006).

The FIRE project at UC Berkeley reports on some of the major difficulties in designing high-tech location systems for the emergency services (Steingart et al.,

2005). Reliability is more important than high resolution or fast updates. Consistent room-level locations every twenty seconds are deemed more useful than finer resolution updates with higher probability of error. And the firefighters must be able to customise and service the equipment themselves to some extent. All this is key to acceptance of new technologies.

2.7.2 Key properties

There have been many efforts over the past decade to help emergency responders navigate in low visibility. Some devices have been produced commercially while others only exist as prototypes, and a few concepts have been described but not implemented. We identify the following criteria that are particularly relevant for designing and comparing different localisation and navigation systems for emergency response.

Primary function *Localisation* determines where the teams are within a structure. *Tracking* records how they got there. *Navigation* shows the teams how to reach a target location (without necessarily knowing exactly where they are). For instance, a flashing beacon provides navigation support without localisation, whereas a number displayed on an office door provides localisation only. A location combined with a correctly oriented floorplan can provide navigation support.

Quality of information Localisation systems are typically characterised by the quality of the information they provide. Researchers often compare accuracy (or resolution), precision (or consistency), and update rate, for different algorithms or systems. These reflect the level of detail and the reliability of a system. Some systems provide reliable location estimates with a lot of detail (good accuracy) whereas others only give coarse locations (poor accuracy). The estimated locations can be

consistent with each other over time (good precision) or can vary to some degree between measurements (poor precision). Navigation systems are more difficult to evaluate than localisation systems without a full trial because of the influence of the display and the behaviour of the user.

Amount of information and flexibility The usability of a system is heavily affected by how much information it provides and how well it can be adapted to different situations. Providing as much information as possible to the user allows them to make their own decisions, but this flexibility sometimes comes at the cost of increased mental workload, whereas a system that filters information, or even makes decisions for the user, will most likely not adapt to unexpected circumstances.

Technology Much of the electronic equipment used by firefighters today is relatively low-tech. High-tech systems tend to be more fragile, more complex to use, and require training, although in some cases the complexity is masked behind a simple and intuitive interface. There is a danger that if the internal workings of the system are not properly understood failure may go unnoticed. Devices containing sensitive electronics are vulnerable to high temperatures and moisture, and must be designed to withstand these conditions (Donnelly et al., 2006).

Components Systems can also be classed according to their number of separate parts, their size and weight. This is particularly relevant when they need to be carried into a building or deployed at the scene. An indoor location system based on a wireless sensor network could consist of a network of tens or hundreds of sensor nodes combined with several body-worn sensors and a small computer. This contrasts with a single self-contained thermal imaging camera for instance.

Deployment and prior knowledge Some systems must be preinstalled in a building in the same way as smoke detectors or sprinklers (Steingart et al., 2005). Others can be installed rapidly in strategic locations upon arrival at the scene of a fire, either outside the building (Graham-Rowe, 2007) or inside by a dedicated team (Renaudin et al., 2007). Yet others are deployed *implicitly* by the search teams themselves as they carry out their mission, e.g., the safety rope or a trail of sensor nodes (Klann, 2009). Self-contained systems such as PASS devices that firefighters carry with them require no deployment at all. For some location systems a digital floorplan is required (Walder et al., 2009).

Limitations All these systems are likely to fail under certain conditions. Some devices will simply cease to work, others may be able to work in a degraded mode and yet others may fail silently and continue to provide incorrect information. One major cause of failure is the coverage which can be limited by the range of a particular signal such as radio, ultrasound or light, or by the number of devices deployed.

Additional features In some cases, a system will provide extra information in addition to location or navigation, for instance by providing reliable radio communication or by monitoring the environment inside the building in real time.

2.7.3 Discussion of systems

The different approaches to localisation mentioned throughout this chapter each have their own strengths and weaknesses when it comes to emergency response. The best solution may be a variation of one of the techniques, or a combination of several. We now describe a number of key research prototypes and commercial systems which use these techniques in the context of emergency response. Table 2.1 summarises the characteristics of the different systems.

SmokeNet Researchers at UC Berkeley have developed SmokeNet (Wilson et al., 2007), a preinstalled sensor network which tracks firefighters in a multistorey building. Sensor nodes installed in each room and approximately every ten metres along corridors provide room-scale location accuracy. Additional sensor nodes monitor smoke and temperature, and relay data to the command post. Colour-coded LEDs show occupants which escape routes are safe. The FireEye display mounted inside each firefighter’s face mask displays a floorplan and short text messages from the command post. The incident commander uses the electronic Incident Command System to see the locations and health status of firefighters, and the status of the smoke detectors. Independently, at Carnegie Mellon University, researchers have used robots to autonomously map the positions of radio (Kantor et al., 2003) or ultrasound (Djugash et al., 2006) beacons. This map can then be used to monitor the progress of a fire and to track firefighters in a similar way to SmokeNet.

LifeNet The LifeNet concept developed by Klann (2009) is designed to provide the functionality of the traditional lifeline (or search rope). It consists of beacons and a wearable device that senses nearby beacons and shows navigational guidance on a head-mounted display. A device attached to the firefighter’s breathing apparatus drops the beacons automatically at appropriate intervals. These form a trail of “breadcrumbs”¹⁷. Each beacon acts as a waypoint to guide the firefighter in either direction. Trails deployed by different firefighters combine to offer alternative escape routes, and loops create shortcuts instead of becoming a trap. The challenge is to present concise and clear information to the firefighters despite the inaccuracies in detecting the direction of the beacons. We collaborated with Klann and his colleagues in this project but the implementation of algorithms and sensors and the identification of the challenges is our own work. We describe a demonstrator for this system in chapter 3.

¹⁷This may not be the best choice of words. In the fairy tale of *Hansel and Gretel*, the breadcrumbs were eaten by birds and the children got lost. White pebbles, however, provided more reliable markers.

Relate Trails The Relate Trails project (Fischer et al., 2008) provides navigation assistance by displaying an arrow on a head-mounted display to help a person retrace their path. The person drops ultrasonic beacons on the way in, and the system uses these to correct PDR position and direction estimates on the way out. Absolute positions may be inaccurate due to PDR drift over long distances but navigation only relies on the position of the user relative to the closest beacons. The use of PDR in addition to beacons allows the system to function to some extent even if beacons are destroyed or out of range. This concept is developed in chapters 5 and 6.

Pathfinder The Pathfinder system produced by SummitSafety¹⁸ consists of a handheld tracker, and beacons which transmit powerful ultrasound pulses. Firefighters can use the tracker to locate a beacon placed at the exit while rescue teams can use it to locate a beacon transmitting on a different frequency worn by a firefighter in distress. Ultrasound waves are blocked by walls but will find a path around corners and under doors; this path can be followed by firefighters. According to the manufacturer, smoke, heat, humidity and audible sounds from the fire do not interfere with ultrasonic waves, and a directional receiver for ultrasound is a lot smaller than for audible sound. The tracker displays the amplitude of the detected signal on a bar graph so a firefighter can locate the direction of a beacon by scanning a 360° circle.

Precision Personnel Location system The Precision Personnel Location system (PPL) (Amendolare et al., 2008) developed at the Worcester Polytechnic Institute uses RF receivers at fixed locations on emergency response vehicles outside the building to track the 3D position of personnel carrying a special transmitter. The RF signals can be used alone to estimate location or they can be used to correct drift in dead-reckoned positions. The dead-reckoning is particularly useful

¹⁸<http://summitsafetyinc.com/> (Accessed 2012.09.24.)

in larger buildings where the RF position estimates are less accurate due to poor signal propagation into the building.

Flipside RFID A team from the National Institute of Standards and Technology (NIST) investigated how predeployed RFID tags embedded in the building could be used to correct PDR (Miller, 2006). They call this the *flipside* of RFID because, unlike typical RFID systems, the tags are static and the mobile reader is worn by the firefighters. The range of the reader and the distance between tags are the key parameters. A long range will only give approximate locations but a short range will miss tags.

Map matching with RFID The drift in PDR position estimates can be corrected by using information from floorplans when these are available. A team from EPFL asks the first team of firefighters to identify doorways by placing an RFID tag on the frame as they pass through (Renaudin et al., 2007). As each tag is placed the location system adjusts the PDR position estimate based on the position of the nearest doorway on the floorplan. The system corrects the orientation estimate based on the direction in which the doorway will typically be crossed. Following teams wear an RFID reader which detects the tags deployed by the first team so the system can correct the position and orientation estimates in the same way.

Map matching with particle filters Researchers from the WearIT@Work project also use floorplans to ensure that successive PDR position estimates do not pass through walls (Widyawan et al., 2008). A particle filter keeps track of thousands of different position and orientation estimates (the particles), and each one is weighted according to how well it fits with the inertial measurements. Particles that pass through walls are eliminated and replaced by plausible ones. The map filtering method works with building outlines but benefits from more detailed floorplans. Woodman and Harle (2008) at the University of Cambridge use maps

which also include vertical positions to represent stairs. Their particle filter uses these 2.5-dimensional maps to track locations over several floors and improve PDR estimates even further.

HeadSLAM HeadSLAM (Cinaz and Kenn, 2008a) combines PDR with readings from a laser scanner mounted on a helmet to produce a map. The scanner detects the direction and distance of obstacles such as walls. The map produced resembles an actual floorplan showing corridors, rooms and doorways. This idea is based on Simultaneous Localisation and Mapping (SLAM) from robotics where a robot gradually builds a map of its environment and keeps track of its current position on the incomplete map (Hähnel et al., 2003). SLAM can be very effective when a robot is allowed to repeatedly scan the environment but it is unclear how well it would perform for a pedestrian in an emergency. We believe our work in chapter 5 offers a more realistic and robust solution by abandoning the head-mounted scanner in favour of foot-mounted sensors and artificial landmarks.

In general we see a tradeoff between systems that provide high quality location information and those that are easy to deploy. Table 2.1 gives a summary of the characteristics of the reviewed systems, while figure 2.2 shows the dependency of good quality location information on preinstalled infrastructure and prior knowledge of the environment. Systems such as PDR, which require little deployment or prior knowledge of the area, tend to be either unreliable or inaccurate. But although preinstalled systems work well under favourable conditions, they cannot be relied upon in a disaster and may not be present at all in many locations.

2.8 Conclusion

We have seen how localisation research in different fields uses a variety of approaches and utilises assumptions specific to their applications. In general, there

Table 2.1: Summary of location support systems for emergency response.

Name	Function		Technology	Deployment	Floorplan	Components	Limitations	Bonus
	Tracking	Navigation						
Available								
Lifeline	Distance	Yes	Rope, knots	Implicit	No	Rope, clips	Limited length, tangled, trapped	No
Torch	No	Yes	Light	Strategic	No	Torch	Obstacles, thick smoke	No
PASS	No	Yes	Alarm	No	No	PASS device	Sound masked, direction difficult to determine	No
PathFinder	No	Yes	Relative ultrasound direction	Strategic	No	Wearable beacon, exit beacon, hand-held tracker	Limited functionality, approximate direction of beacon only	No
Commercial indoor location systems	Yes	No	UWB or ultrasound	Preinstalled	Optional	Sensors in building, wearable tag	Sensitive calibration, loss of connectivity and power	No
Prototypes								
PPL	Yes	No	RF ranging, inertial sensors	Strategic	Optional	Multiple receivers outside, mobile transmitter	Metal structure, large buildings	No
SmokeNet	Yes	No	RF fingerprints	Preinstalled	Required	1 beacon per room and every 20m	Changes in environment	Environment monitoring, communication
LifeNet	Distance	Yes	Relative ultrasound direction	Implicit	No	Beacons every few metres, wearable sensor	Beacons moved or destroyed	Environment monitoring, communication
PDR	No	Yes	Inertial sensors	No	Optional	Shoe-mounted sensor	Drift, unpredictable error	No
Map matching with particle filter	Yes	No	Inertial sensors	No	Required	Shoe-mounted sensor	PDR drift	No
Map matching with RFID	Yes	No	Inertial sensors, RFID	Strategic	Required	Shoe-mounted sensor, wearable RFID reader, RFID tags	PDR drift	Posture monitoring
Flipside RFID	Yes	No	Inertial sensors, RFID	Preinstalled	Required	Shoe-mounted sensor, wearable RFID reader, RFID tags	PDR drift	No
Relate Trails	No	Yes	Inertial sensors, relative ultrasound direction	Implicit	No	Shoe-mounted sensor, beacons, wearable sensor	Beacons moved, PDR drift	Environment monitoring, communication
HeadSLAM	Yes	Yes	Inertial sensors, laser range scanner (relative distance and direction)	No	No	Head-mounted inertial sensors and scanner	Scanner fails in low visibility, PDR drift	Environment monitoring, communication

is a trade-off between reliability, resolution and ease of deployment. This is particularly relevant when it comes to emergency response where each of those requirements is important. Time is precious, so systems which require no deployment, such as inertial dead-reckoning and vision-based SLAM, appear attractive. But these systems may not be robust enough for safety critical applications in harsh environments. Methods from industrial asset tracking and military target tracking have some benefits in terms of reliability, but they rely on complex infrastructure. The resolution of the location estimates is a key consideration because responders must be able to find a doorway or a person in near zero visibility. Some commercial indoor location systems can provide the required level of accuracy but, again, they require pre-installed and calibrated infrastructure. WSN researchers working on localisation strive to design systems that will work reliably in harsh environ-

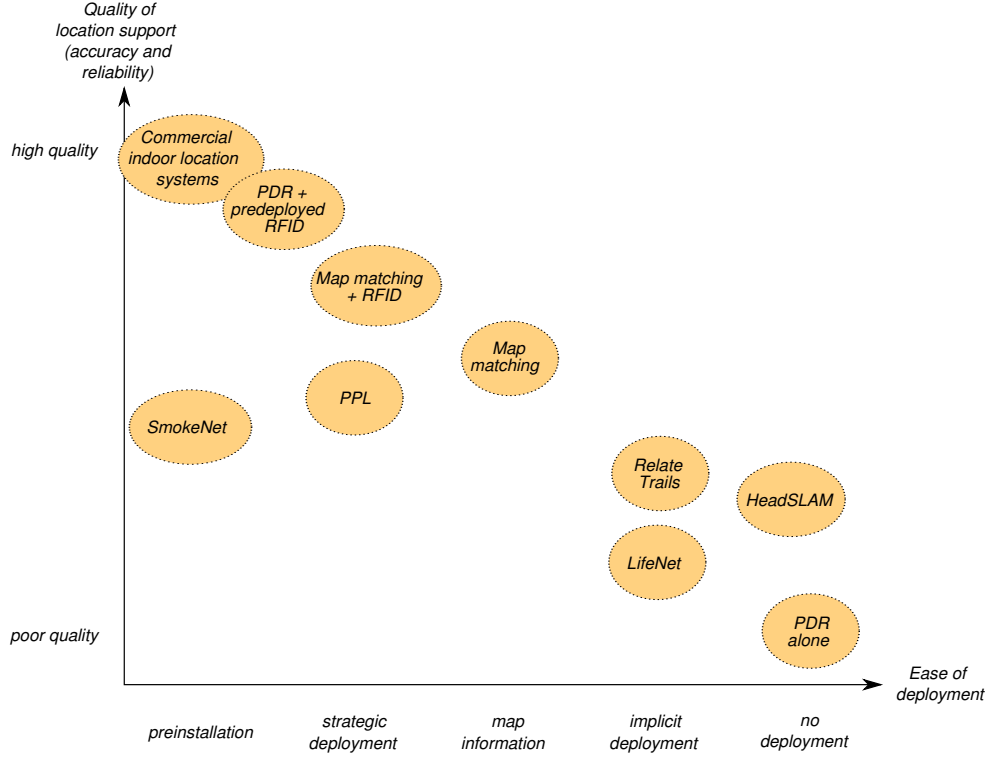


Figure 2.2: Comparison of different location systems according to the ease of deployment and quality of location support. Increased reliability and accuracy comes at the cost of more deployment or prior knowledge of the environment.

ments with minimal human intervention, and, as in other areas, there is an effort to improve resolution; however, wireless sensor networks often achieve this by using large numbers of nodes and aggregating measurements over a wide area and a long time. Existing research into localisation for emergency response attempts to resolve these conflicting requirements by using elements from these different areas.

We find that, although the research community and industry are aware of the demand for tracking and navigation support for firefighters, none of their proposed solutions adequately address the issues. Systems that provide high quality location information in terms of reliability and granularity still tend to rely on pre-installed infrastructure or prior knowledge, such as floor plans, while systems based purely on wearable sensors cannot currently guarantee sufficient tracking accuracy for the duration of an emergency intervention. One avenue of exploration for researchers is therefore the coupling of dead reckoning with other localisation techniques, without substantially increasing the deployment effort.

In the following chapters of this thesis, we look more closely at a selection of existing technologies and algorithms, and see how they can be used for pedestrian tracking in uninstrumented and unknown environments. In chapter 3, we explore the feasibility and practical application of relative positioning, that is the estimation of positions of objects relative to each other, outside of any absolute coordinate frame (such as a map). We illustrate this by looking at a method from the field of WSN research using ultrasonic ranging, and propose a variation which allows us to track sensor nodes in real time. In chapter 4, we show that we can use body-worn sensors to measure movements from a given starting point, and provide position estimates. We cover the topic of inertial pedestrian dead-reckoning by giving a reference implementation and describing some of the requirements to make it work as well as possible. Then, in chapter 5, we learn how to create a map and track a person in that map without any prior knowledge of the landmark positions. We achieve this by combining the sensor network approach and the inertial navigation approach to provide simultaneous localisation and mapping (SLAM), a technique used in the field of autonomous robots. Finally, chapter 6 explores how well this SLAM method might work in practice, when used in a search and rescue mission, by taking into account the deployment of the sensor nodes, the inaccuracies of the algorithm, and the user interface.

Peer-to-peer ultrasonic measurements for relative positioning

In the previous chapter, we remarked that many localisation systems estimate positions in an absolute coordinate system. GPS, for instance, uses a global geographic coordinate system (WGS84), while some of the other solutions require coordinates to be referenced to a floorplan. This is not suitable for indoor localisation in a building for which maps are inaccurate or unavailable. In this chapter, we examine the feasibility of relative localisation, that is localisation of objects solely with respect to each other, without requiring an external frame of reference. This is an important step towards providing navigation support for emergency response.

Some of the work in this chapter is based on localisation techniques from wireless sensor networks, but the applications include localisation in smart environments and navigation assistance for emergency response. Our various studies and demonstrators have in common that they are all built around sensor nodes that measure distance and angle (also called *range* and *bearing*) to each other using ultrasound. We call these measurements *peer-to-peer* because each one is taken between two nodes without requiring any external synchronisation or calibration

with respect to another device or reference point. The measurements can still be coordinated and collected by a central device for convenience but the measurements themselves are peer-to-peer. The sensor nodes we use are also wireless and battery-powered. These characteristics of the sensor nodes and the measurement process make these studies relevant to our research into location support in uninstrumented and unknown environments.

In this chapter, we first examine some applications and proof-of-concept implementations which show how relative position measurements between devices can be used in place of more traditional infrastructure-based localisation systems to provide location-aware services. These initial studies uncover a number of practical limitations which restrict the use of these systems in the real world. We then propose an algorithm which addresses some of these issues and allows us to more reliably track mobile devices while maintaining the infrastructure-less nature of the system.

3.1 Background and simple demonstrators for relative localisation

Many research projects aim to provide indoor localisation by instrumenting the environment. In contrast, the Relate project¹ investigated technologies, methods, algorithms, and applications for relative positioning of electronic devices only with respect to each other, independently of the environment. This was driven by the observation that many location-aware services rely more on the positions of devices relative to each other than absolute positions within a building. For instance, a system that teleports your computer desktop to a suitable screen as you move between offices only needs to know how close you are to each available screen and which screen you are facing (Harter et al., 2002). Using only this minimal

¹Project No. 013790, FP6 IST Programme funded by the European Commission

amount of data, the application should also be more robust to change; it should work seamlessly without reconfiguration when screens are shuffled around between offices. At first, the Relate project was concerned with collaborative work, location-aware user interfaces, and interactions with smart environments. These topics are described in the first part of this section and in an article by Gellersen et al. (2008). Later in this section, we describe another application where we provide navigation support to emergency responders using devices from the Relate project.

3.1.1 Hardware: ultrasonic sensor nodes

Several prototype devices were developed by our colleagues in this project in order to explore relative positioning. The first devices use a circular array of infrared receivers and transmitters to measure range and bearing to each other (Krohn et al., 2005). The following generation of devices are the USB *dongles* described by Hazas et al. (2005). Each dongle has three ultrasonic transducers and connects directly to the USB port of a laptop or PDA. Later work made use of battery-powered *dots* which can be attached to various objects in the environment without needing to be connected to another device. Finally, *bricks* are more typical sensor nodes, battery-powered, with four transducers each and an optional USB connection for data logging. Figure 3.1 shows these three types of ultrasonic sensor nodes. They are all compatible with each other. For the work in this thesis, we used the hardware as provided but made some minor improvements to the sensor firmware.

Each of these devices is built around a Particle Computer (Decker et al., 2005) and its associated AwareCon network protocol stack. Measurements are taken when a transmitter sends a *trigger packet* over the RF channel and simultaneously emits an ultrasonic pulse. This allows the receivers to measure the time-of-flight (inferring the distance) and estimate the angle of arrival (inferring the bearing to the transmitter). If only one of the transducers detects the ultrasonic pulse, the bearing is estimated as the multiple of 90° corresponding to that transducer. If the

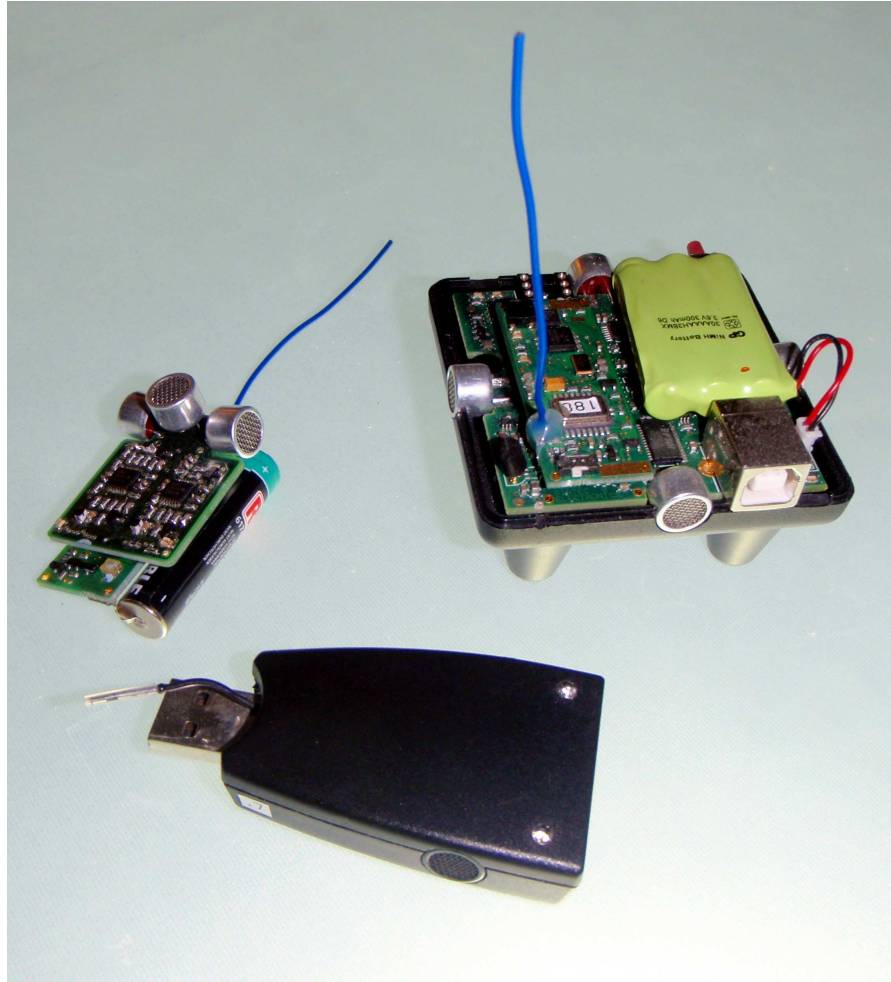


Figure 3.1: Ultrasonic devices from the Relate project — clockwise from right: *brick*, *dongle* and *dot*.

pulse is detected by two or three transducers, the bearing is estimated by linear interpolation based on the pulse amplitude

$$\phi = \phi_{max} + \frac{\pi}{4} \frac{a_{left} - a_{right}}{a_{max}}, \quad (3.1)$$

where the angle ϕ_{max} is the multiple of 90° corresponding to the transducer having received the pulse with the greatest amplitude a_{max} ; a_{left} and a_{right} correspond to the amplitude detected on this transducer's left and right neighbours respectively.

The receiver measures the local temperature in order to more accurately estimate the speed of sound but variations in the environment can introduce error into the measurements. The largest errors are due to reflections, so some simple heuristics are applied to detect and eliminate some of these bad measurements. For instance, the firmware flags measurements where the transducer measuring the shortest time of flight is different from the transducer receiving the strongest signal. Using similar principles, each measurement is assigned a quality rating between zero and four. Range and bearing measurements where three transducers detected consistent pulses (a strong pulse on the central transducer, and weaker pulses with a short delay on both lateral transducers) are rated four and are the most reliable.

We encountered a particular problem when we started working on larger scale experiments using the more powerful pulses generated by the Relate bricks. Each device waits approximately thirteen milliseconds between transmission of successive ultrasound pulses. During this time, the pulse has travelled around 4.5 metres. If the receiving device is more than 4.5 metres away, it will give up listening for the first pulse after thirteen milliseconds and start listening for the second one. Then the first pulse arrives and the receiver underestimates the distance by 4.5 metres. The second pulse is assumed to be a reflection and ignored. The solution we adopted was to discard measurements with a short estimated distance but a

weak amplitude. A better solution requiring more advanced hardware would use variable gain amplification where the gain increases with time.

3.1.2 Non-linear regression

One of the first application scenarios for relative positioning in the Relate project was collaborative work (Kortuem et al., 2005). A group of co-workers sits around a table, each with their own laptop or PDA. Each device is equipped with an ultrasonic Relate dongle and software to estimate the relative positions of each of the other devices. This position information is then used to populate *spatialised widgets* which display a map of neighbouring computers, or allow users to select a recipient for an instant chat message or a file transfer from a drop down list sorted by distance.

The algorithm used for this system operates as follows and is also described by Krohn et al. (2005).

1. A batch of measurements is recorded over a few seconds. Ideally, there should be at least one range and bearing measurement between each pair of sensor nodes.
2. Positions and orientations for all sensor nodes are initialised using a graph tracing algorithm which attempts to successively place each node in a graph using only the best available measurements.
3. A Levenberg-Marquardt non-linear regression minimises the differences between the estimated positions and orientations, and the measured values. We use a slightly modified version which ensures that angle-wrap is handled correctly, e.g., an error of 361° is the same as an error of 1° . The minimisation is run several times in an attempt to eliminate outliers; each time, the measurement with the highest residual is removed.

Hazas et al. (2005) show that this method works well to estimate the positions and orientations of stationary devices in a plane (such as a tabletop). Since the algorithm processes measurements in batches and does not use a history of position estimates, it will continue to work just as well after devices are moved to different positions. It may also work, albeit with a lower accuracy, while a single device moves slowly and the others remain at fixed positions. But it is not able to track multiple moving devices or a fast moving device due to the inherent measurement errors which occur for mobile devices, and to the fact that measurements within a batch will have been taken at different positions over a few seconds. The *simultaneity assumption* is broken (Welch and Bishop, 1997). The non-linear regression does not scale well with the number of nodes. More nodes mean more unknowns to estimate, and larger batches of measurements which take longer to accumulate and to process.

3.1.3 Using individual measurements

We also use the range and bearing measurements from the Relate sensor nodes to create two demonstrators designed to help a person navigate in their surroundings. The first helps a person identify what parts of a smart environment they can interact with, the second provides navigation support for a firefighter. The sensor nodes from the Relate project measure angle of arrival as well as distance; this is essential for orienting oneself. Both these applications have in common that they use only single measurements and do not perform any kind of filtering such as the non-linear regression described above.

Relate Gateways: an interface for spontaneous interaction

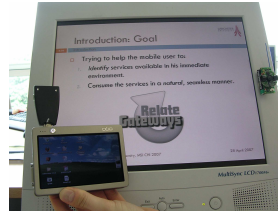
We consider the scenario where a user with a laptop, PDA, or smartphone walks around a building, and wants to interact seamlessly (and wirelessly) with services

available in the environment (Guinard et al., 2007; Gellersen et al., 2008) For example, they may want to show a colleague a presentation on a large public display rather than the tiny phone screen, or print a document from their laptop when they happen to pass a printer on their way to a meeting. The Gateways interface displays icons around the edge of the screen of the mobile device; each icon represents a service available in the environment, such as a printer, a display or a loudspeaker. These icons, the *gateways*, are located around the screen in positions corresponding to the position of the device in the real world in order to make the interaction more intuitive (fig. 3.2a).

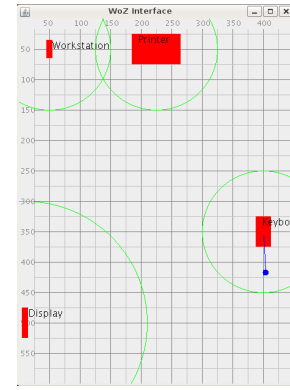
We designed an additional ultrasonic sensor node called the *dot* to make this scenario possible. The dots have a much smaller form factor than the dongles, and no USB connection, but they are compatible with the Relate USB dongles (fig. 3.2b). A dot is attached to each service in the environment but not connected in any way; each dot is battery-powered and wireless. The dots are programmed to each transmit in turn, and to broadcast the type of service and the URL which identifies and gives access to this particular service. Mobile devices listen via their ultrasonic USB dongle, identify, and locate nearby services which are then displayed as icons in suitable positions around the edge of the screen. In principle, this system works, but in practice the measurements are not sufficiently accurate or stable, and this makes the interface difficult to use as the gateways jitter. Although the concept was demonstrated with some success at several venues (Fischer et al., 2007a,b), we more frequently used a *Wizard of Oz* setup instead of the sensor-based localisation system. A researcher (the “wizard”) sits discreetly in a corner of the room and clicks on an office floorplan to indicate the location and orientation of the user (fig. 3.2c). This updates the display on the user’s device more reliably than the sensor-based system, and allows us to study the user interface independently of the measurement errors.



(a) A handheld computer showing three *gateways*: a shared screen to the left of the user, a printer to their right, and a keyboard located behind them to the right.



(b) Part of the demonstration: a handheld computer with its Relate dongle and a shared screen tagged with a *dot*.



(c) The Wizard-of-Oz interface for conducting user studies. Here the user is standing close to a keyboard and facing it.

Figure 3.2: Components of the Relate Gateways prototype.

Relate Trails: supporting firefighter navigation

In collaboration with the wearIT@work project², we developed software that uses a trail of Relate devices as a *virtual lifeline* to assist a firefighter in finding their way along a path (Klann et al., 2007). The virtual lifeline is intended as a replacement for the physical ropes that are sometimes used by firefighters. It is deployed behind them as they advance into a building, and guides them back to the exit when they need to retreat.

The firefighter's boots include a Relate ultrasonic sensor node that transmits and receives ultrasound pulses from four transducers attached around the sole (fig. 3.3). The sensor nodes deployed on the ground have similar hardware and all communicate wirelessly. Each time a new node is dropped and turned on, it detects other nearby nodes and is added to this firefighter's virtual lifeline. Each node in the lifeline keeps track of its distance to the exit (the first node) by accumulating its measured distance to the previous node with the previous node's distance to the exit. The nodes periodically transmit heartbeat messages and update the distance

²Project No. 004216, FP6 IST Programme funded by the European Commission.



Figure 3.3: Boots equipped with ultrasonic sensor nodes. The transducers are located around the sole.

to their neighbouring nodes. When a node has not transmitted for a while, or has moved too far away, the lifeline registers that this path to the exit is no longer available. New nodes are only dropped when a firefighter has moved a certain distance away from existing nodes. When multiple lifelines are close to each other, the nodes from each line detect it and register that an alternative path to an exit is available. The firefighter's transmitter regularly sends pulses to determine which lifeline nodes are nearby. The lifeline nodes then respond by sending a pulse and telling the firefighter how far they are from the exit. Their pulse allows the firefighter's sensor node to measure which direction each node lies in. The firefighter is guided to the best node by an arrow on their head-mounted display. When they move closer to this node, another node closer to the exit is detected and the direction of the navigation arrow is updated accordingly. They are thus guided step-by-step towards the exit.

This demonstrator worked in the lab but only poorly when we showed it at a conference. Once again, the problem was unreliable measurements: incorrect angles caused the arrow to swing erratically, incorrect distances caused the system

to miss the nearest node, and delayed radio messages caused nodes to timeout and be removed from the line. These issues were traced back to several sources: radio interference from other demos, ultrasonic interference from a noisy environment, reflection of ultrasonic pulses off obstacles, weak pulse due to unfavourable orientation of devices, measurement collisions between multiple nodes. Only the latter could potentially be addressed through more careful coordination of the measurement protocol, but all the other issues would require substantial hardware and low-level software development. We believe these changes to be achievable with sufficient resources.

3.2 Real-time relative positioning of mobile nodes in sensor networks³

In the previous section, we discussed the use of ultrasonic sensor nodes to enable location-aware collaboration and interaction with services in a smart environment, and to support navigation for firefighters. The concept of relative localisation with no infrastructure was shown to work in principle, but the localisation algorithms we used and the quality of the measurements were inadequate for tracking mobile devices in real-time. Not only are the particular algorithms incapable of producing good position estimates while a device is moving because the simultaneity assumption is broken (Welch and Bishop, 1997), but they also take several seconds to stabilise after movement has stopped. In our opinion, this is not good enough for a system with a live user interface, as would be the case for a navigation system.

In this section, we focus on a more suitable algorithm which gives good results even when many devices are moving. This mobile scenario occurs when one or more

³This section contains unpublished work conducted by Carl Fischer with Matt Fisher and Carl Ellis under the supervision of Mike Hazas at Lancaster University. Matt Fisher wrote the initial software demonstrator in Java; Carl Ellis implemented the MDS-MAP initialisation; we coordinated the work, conducted the experiments, and reimplemented and refined the algorithm.

people equipped with sensors (either attached to their bodies, or on their phones) are trying to navigate an unknown area. In order to develop, test and compare our algorithm to other methods, we use a generic wireless sensor network (WSN) rather than one designed to support a particular application. Hence, we use the approach and the vocabulary of WSN research rather than those of ubiquitous computing or pedestrian tracking. WSNs are appealing to us because they are composed of small and easy-to-deploy sensor nodes which often work in a decentralised fashion, and the embedded algorithms for communication, localisation and data collection are typically able to initialise and reconfigure themselves as required. These characteristics are a good match for the *uninstrumented* environments we are considering. We use measurements from a real-deployment of sensor nodes but the processing is done offline in Matlab. Online processing is feasible with a few changes discussed at the end of this chapter.

3.2.1 Introduction

We argue that Kalman filtering is a positioning solution more generally suited to the localisation of sensor nodes than other algorithms, because of its accuracy, robustness to measurement noise often encountered in WSNs, real-time tracking capability of multiple mobile devices, and low computational and communication overheads which scale linearly with the number of neighbours of each device. We demonstrate this argument by a deep performance analysis of a Kalman filter which uses range, and optionally bearing, measurements between sensors to estimate their locations and orientations, including when all sensors are mobile.

We characterise the filter using data taken from our Relate sensor nodes (section 3.1.1) which transmit and receive ultrasound pulses; they use time-of-flight to estimate the range between the nodes, and the relative amplitude of the received pulse at different transducers to estimate the angle of arrival (or bearing). There

are a number of other methods and technologies which have been used to measure range and bearing. For example, range can be estimated using the received signal strength or the round-trip time of a radio transmission, and bearing can be measured with a directional antenna or an antenna array. Alternatively, a camera can be used to determine the range and bearing to a visual marker. Regardless of sensing modality, this tracking algorithm can be applied in any sensor network where node-to-node range, and optionally bearing, can be measured. Other types of measurements such as speed (Amundson et al., 2008) can easily be incorporated into the system by defining a suitable measurement model (see equation 3.4).

In previous WSN literature, the Kalman filter (and more generally, Bayesian filtering) has been used to track a single mobile node moving within a field of static neighbours, using a centralised algorithm (Savvides et al., 2002; Taylor et al., 2006). Our core argument is that the Kalman filter is a strong candidate for generalised WSN positioning, with the following advantages:

- General algorithm with extensive supporting literature from other fields.
- Configurable and extensible to incorporate a variety of sensing modalities, measurement types, and node mobilities.
- Can be distributed across the nodes thanks to low computation and communication requirements.
- Scales well to large multihop networks, as each node needs only knowledge of measurements to its immediate (single hop) neighbours.
- Location accuracy and robustness to sensor noise is better than or equal to other WSN localisation algorithms in this context.

3.2.2 Related work on localisation in wireless sensor networks

There is a large amount of literature on sensor network localisation, primarily focusing on localising static nodes in multihop networks, where not all nodes are within measurement range of one another (Niculescu and Nath, 2003; Shang and Ruml, 2004; Moore et al., 2004). These and similar algorithms have been comparatively evaluated for different network densities, measurement ranges, and topologies (Langendoen and Reijers, 2003). Whitehouse and Culler (2006) have shown how algorithm performance can be severely degraded by raw measurement error characteristics which deviate from the commonly-used Gaussian “noisy disk” model. To evaluate algorithm performance, they recommend using simulations which draw samples from error distributions compiled from real-world ranging data. Our experimental methodology and analysis follows this strategy.

A few researchers have also used bearing measurements as well as ranges. Chintalapudi et al. (2004) describe a localisation algorithm based on an iterative optimisation using blocks of both range and bearing measurements. They first show that by using bearing measurements with small errors of a few degrees, sparser networks with fewer anchors can be localised adequately. Then they show that even when only rough sectoring information is available (e.g., 45° sectors) they are able to get better results than with ranges alone by including the sectors in the initialisation phase of the algorithm. They achieve good results but their algorithm requires anchor nodes with known positions and operates on batches of measurements, making it unsuitable for real-time tracking of mobile devices.

Since our method is specifically designed to deal with mobile as well as static nodes, we concentrate here on prior sensor network localisation algorithms involving mobile nodes. Park et al. (2008) use the expression *moving-baseline localisation* to track nodes “operating in the absence of a fixed reference frame”. Rather than

solving a series of static localisation problems they determine the parameters of the nodes' trajectories which are assumed to be piecewise linear. Based on real range measurements between a pair of ultra-wideband radio nodes, they apply a distance-dependent Gaussian model for their simulations, with a 5% chance for "large error" which is drawn from a uniform distribution from zero to ten metres. They show that the algorithm can perform tracking even when all nodes are simultaneously mobile. However, it is unclear how it performs with more aggressive types of noise (systematic over- and under-ranging), observed in the multinode ultrasonic deployments by ourselves and others (Whitehouse and Culler, 2006).

The Monte Carlo Localisation Boxed method described by Baggio and Langendoen (2008) uses a particle filter to locate a set of mobile nodes. Their method requires some of the nodes, called anchors, to have known positions. This is a "range-free" method which relies on knowledge of whether two nodes are in range of each other (connectivity). It is designed to work well with mobile nodes, and they evaluate their method via simulation. The advantage of a particle filter is that it can model arbitrary measurement error distributions rather than simply Gaussian ones, and it can model a non-linear relationship between measurements and position estimates instead of the linear approximation used in an extended Kalman filter. However, this comes at a much higher computational cost required to process the hundreds of particles.

Galstyan et al. (2004) employ bounding box constraints to estimate the locations of static nodes, as they sense a mobile node moving among them. Taylor et al. (2006) build on this idea, but incorporate live tracking of the mobile node, as well as refining estimates for static nodes. They employ a Bayesian filter operating on range-only measurements, and evaluate its accuracy and convergence using a single data trace from each of three deployments. Our proposed Kalman filter-based method differs from theirs in two important ways: (1) we do not differentiate between static and mobile nodes, all our nodes are identical in terms of hardware

and software, (2) we do not rely on collecting batches of measurements and are therefore able to localise fully mobile networks. Their three data traces are used to show how well, on average, the positions of static nodes are refined, as a mobile node moves among them. The mobile node tracking accuracy is not reported, since no dynamic ground truth capture was in place. We plot dynamic tracking accuracy distributions for increasing proportions of mobile nodes. Savvides et al. (2002) also use a Kalman filter formulation in the centralised version of their n-hop multilateration algorithm but it is used to process blocks of range measurements between static nodes.

By contrast, our Kalman filter is a “single constraint at a time” (SCAAT) realisation, most similar to the dynamic tracking and autocalibration method used by Welch and Bishop (1997). Cameras on their mobile HiBall tracker device detect the bearing to ceiling-mounted LEDs; the SCAAT filter uses these individual measurements to compute the location and orientation of the HiBall with a Kalman filter, while concurrently refining the LED position estimates.

Work by Kusy et al. (2007) and Amundson et al. (2008) tracks a single mobile robot by processing Doppler shift velocity measurements with a Kalman filter. The sensing technique is based on radio signals which have a much wider coverage than our ultrasonic pulses but the location errors are of a similar proportion to ours, given the scale of the experiment. They use a maneuver detection algorithm to improve performance when the robot changes direction and the constant velocity assumption is broken. Their Kalman filter implementation can be seen as a specialised version of the more general algorithm we present below. Their implementation is tailored to suit their scenario of a single mobile node moving through a field of stationary nodes with a relatively low measurement rate.

3.2.3 Relative Kalman Filtering

The Kalman filter is an iterative Bayesian method for estimating the state of a system given a series of observations. In our particular implementation, it estimates the locations of the sensor nodes relative to each other, given range and bearing measurements between them. It is particularly attractive for tracking problems because it can process single measurements at a time and provide location estimates in real-time, unlike the batch methods described above where the system must collect a number of measurements before being able to run the localisation algorithm. The design we have chosen is suited to lightweight sensor nodes because each sensor only locates itself and its immediate neighbours, and is not required to store any other information than those estimated locations. For an overview of the Kalman filter applied to localisation and navigation we refer the reader to Groves (2008), and for a more formal derivation of the Kalman equations to Simon (2006).

The defining features of our specific implementation, which we use to demonstrate the advantages of Kalman filtering for the localisation of wireless sensor nodes, are as follows:

- Range and optionally bearing measurements are used to locate static nodes and to track mobile nodes in real-time.
- Mobile nodes can be located in real-time relative to each other, even when all nodes are mobile.
- No anchor nodes (nodes with known positions) are necessary. All nodes are located within the network, within one coordinate system. As there are no local neighbourhood coordinate systems, no “patch-and-stitch” step is necessary to merge to a global, network-wide coordinate system (Whitehouse and Culler, 2006).

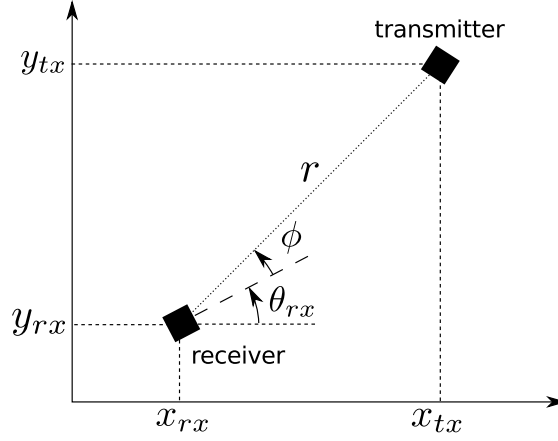


Figure 3.4: Notations used for coordinates, orientations and measurements. x_{rx} , y_{rx} and θ_{rx} are the coordinates and orientation of the receiver. x_{tx} and y_{tx} are the coordinates of the transmitter. r and ϕ are the range and bearing measurements.

- Accuracy is better than previously proposed localisation algorithms which use range and, optionally, bearing, and the required computation is less. Our method is suitable for distribution on lightweight sensor nodes.

Typically, a Kalman filter tracks the position of a single object in an absolute coordinate system, but we use it to track the position of multiple objects relative to each other. In our case, the state of the system consists of the 2D locations and orientations of the sensor nodes. An observation, or measurement, consists of the range and bearing measured by the receiving device with respect to the transmitting device. We denote the state of the receiving device by $state_{rx} = (x_{rx} \ y_{rx} \ \theta_{rx})^T$. The state of the transmitting device uses the subscript tx . Measurements are denoted by $meas = (r, \phi)$. These variables are illustrated in fig. 3.4.

In addition to the estimated state, the filter maintains an error covariance matrix which represents the uncertainty of the state estimate. The 3×3 error covariance matrices are denoted by P_{rx} and P_{tx} respectively. We will sometimes refer to the state simply as the *position*, although it also includes the orientation.

Instead of working with a single large state vector containing coordinates of all devices, our filter constructs a state vector on the fly with only the states of the two devices involved in the current measurement. The same principle is applied

to the error covariance matrix. When the filter has finished processing the current measurement, the states and covariance matrix of the two devices are saved. The aggregate state and error covariance are constructed as $state = \begin{pmatrix} state_{rx} \\ state_{tx} \end{pmatrix}$ and

$$P = \begin{pmatrix} P_{rx} & 0 \\ 0 & P_{tx} \end{pmatrix}.$$

The steps to process a measurement are as follows:

1. Construct the state vector and error covariance matrix by combining the previously stored values for the transmitter and receiver involved in this measurement.
2. Predict the current state according to the previous state and the system model. Update (increase) the error covariance accordingly. These are the *a priori* values because they do not take into account the latest measurement. We denote them by $state^-$ and P^- .
3. Update the predicted state with information from the measurement, weighted according to the predicted estimate covariance and the measurement error model. Update (decrease) the error covariance accordingly. These are the *a posteriori* values, denoted by $state^+$ and P^+ .
4. Split the state vector and the error covariance matrix into the part corresponding to the transmitter and the receiver. Store these values.

Initialisation

Initialisation is likely to be application and network dependent. The following discussion relates particularly to our sensor nodes which transmit pulses in a round-robin fashion. In situations where measurements are scheduled differently, or where

some sensors are known to be static and have positions which are known approximately, different initialisation schemes may be more suitable.

When the filter is started, one device is arbitrarily assigned the coordinates $(0, 0)$ and an orientation of 0° . When a device transmits a pulse which is received by another device with a known position and orientation, we can determine the position of the transmitter but not its orientation (eq. 3.2). And when a device with a known position receives a pulse from another device with a known position, we can determine the orientation of the receiver (eq. 3.3). Based on these conditions, our algorithm initialises devices' positions and orientations as soon as the necessary measurements are available. Note that the position estimate of the first device will change as further measurements are processed by the filter; it will not necessarily remain at the origin of the arbitrary coordinate system. A more robust variation on this naive initialisation method is to wait for several measurements between each node pair and take the median.

$$\begin{aligned} x_{tx} &= x_{rx} + r \cos(\theta_{rx} + \phi) \\ y_{tx} &= y_{rx} + r \sin(\theta_{rx} + \phi) \end{aligned} \tag{3.2}$$

$$\theta_{rx} = \text{atan}\left(\frac{y_{tx} - y_{rx}}{x_{tx} - x_{rx}}\right) - \phi \tag{3.3}$$

In a single hop network such as the one we start with, where all devices transmit in round-robin fashion and assuming there is no packet loss, the total initialisation time is the number of devices multiplied by the duration of a measurement. For a multihop network, this is multiplied by the maximum number of hops from the first device. In a multihop network, this naive initialisation method can be slow because devices several hops away from the first device will only be initialised after closer devices have themselves been initialised. Errors also accumulate over multiple hops when this basic method is used. In order to speed up initialisation and

to avoid the inconsistencies that can arise between different portions of a multi-hop network, we investigate the performance of alternative initialisation methods such as a graph tracing algorithm (which applies simple trigonometry to range and bearing measurements to produce rough location and orientation estimates) and MDS-MAP (Shang and Ruml, 2004). These methods are discussed in subsection 3.2.5.

Prediction of the state and error covariance

The prediction of the state is trivial for static devices because their position remains the same between measurements. The covariance matrix for static devices is also propagated trivially as $P_{k+1}^- = P_k^+$, where k is the timestep. For mobile devices, we may use a constant velocity model or a random movement model as discussed in subsection 3.2.3.

At this stage, the filter can predict the range and bearing measurement according to the measurement function h .

$$h(state) = \begin{pmatrix} r \\ \phi \end{pmatrix} = \begin{pmatrix} \sqrt{(x_{tx} - x_{rx})^2 + (y_{tx} - y_{rx})^2} + \epsilon_r \\ \text{atan}\left(\frac{y_{tx} - y_{rx}}{x_{tx} - x_{rx}}\right) - \theta_{rx} + \epsilon_\phi \end{pmatrix} \quad (3.4)$$

$(\epsilon_r \ \epsilon_\phi)^T$ is the measurement noise with covariance R , determined empirically. Our devices provide a quality indicator which we use to estimate the noise covariances for each measurement individually. The quality indicator takes the value 1, 2 or 3, based on the number of transducers at which a sufficiently strong pulse was detected. A pulse received by more transducers tends to have a smaller range and bearing error, and the exact value of R is chosen according to this. We use the 90th percentile value of range and bearing errors (approximately 7cm and 30° when considering all measurements).

Update of the state and error covariance

The measurement model in eq. 3.4 gives the range and bearing as a function of the positions of the devices. The measurement function h is non-linear so we use an extended Kalman filter (EKF) which linearises h around the current state estimate. The EKF uses the matrices of partial derivatives (Jacobians) of the non-linear functions. In particular, H is the matrix of partial derivatives of the measurement function h with respect to the state variables $x_{rx}, y_{rx}, \theta_{rx}, x_{tx}, y_{tx}$.

The Kalman gain K represents the ratio between the uncertainty of the prediction and the uncertainty of the measurement, and is computed as $K = P^- H^T (H P^- H^T + R)^{-1}$.

The gain is then used to combine some of the information from the measurement with some of the information from the prediction as $state^+ = state^- + K(meas - h(state^-))$.

To improve the linearisation, we use an iterated update phase (Jazwinski, 1970). Each iteration calculates H at the current state estimate, then computes the gain K and re-estimates the state until the state estimate converges.

Finally the error covariance P is updated with $P^+ = (I - KH)P^-(I - KH)^T + K R K^T$. This form guarantees that P remains symmetric positive definite (Simon, 2006) and avoids potential instability. R is the covariance matrix of the measurement noise vector $(\epsilon_r \ \epsilon_\phi)^T$ and I is an identity matrix of appropriate dimensions. The state vector and error covariance matrix are then split into their respective components $state_{rx}, state_{tx}, P_{rx}$ and P_{tx} .

Outlier filtering

We define $innovation = meas - h(state)$ as the difference between the predicted measurement and the actual measurement. It can be used to estimate whether

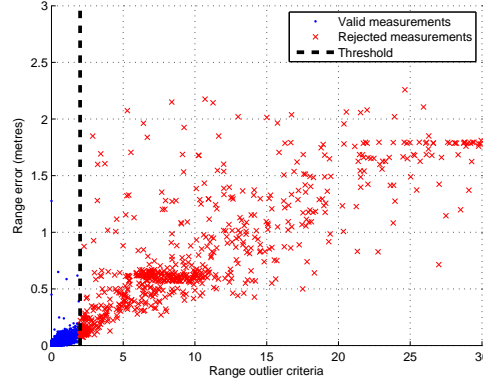


Figure 3.5: Filtering of outliers in a deployment with fifteen nodes. Measurements with high range error are successfully rejected. In this example, 5% of all measurements are rejected as outliers. Eq. 3.5 gives the expression of the criteria shown as the abscissa.

a measurement is an outlier. We discard all measurements for which the ratio of innovation to estimate uncertainty is too high. Every time the filter eliminates an outlier we increase the uncertainty of the corresponding devices so that in the case where the device really has moved or the current estimate is incorrect the filter will eventually update its position instead of discarding all further measurements. This is essential because it allows the position estimates to be corrected when the initialisation is wrong. It is an alternative to techniques such as adding artificial process noise or using a fading memory filter (Simon, 2006, sec. 5.5) which are necessary for a Kalman filter to correctly estimate static values.

Our outlier criterion is highly correlated with the actual range error as illustrated in figure 3.5. By removing all measurements where this criterion is greater than an empirically determined threshold, we successfully remove many of the measurements with high range errors while keeping all measurements with low range errors. The range innovation is the most useful for detecting outliers but the bearing innovation can also be used.

The outlier criterion is computed as follows:

$$range_outlier_criterion = \frac{innovation}{\sqrt{max(P_{rx}(1, 1), P_{rx}(2, 2)) + max(P_{tx}(1, 1), P_{tx}(2, 2))}}. \quad (3.5)$$

Filtering for mobile devices

Mobile device tracking can be accomplished with only minor modifications to the filter specified above. One approach is to model a mobile device's velocity, and assume that it remains constant between measurements. This approximation is almost perfectly valid in our experiments during the straight segments of the robots' paths but not when they are changing direction and reversing. The state and error covariances of each mobile device are augmented to include linear velocity so the state vector for a mobile device becomes $state = (x \ y \ \theta \ \dot{x} \ \dot{y})^T$. The prediction of P now becomes $P_{k+1}^- = AP_k^+ A^T + WQW^T$ where A is the matrix of partial derivatives of the prediction function (the system model) with respect to the state variables, Q is the process noise covariance, and W is the matrix of partial derivatives of the prediction function with respect to the process noise variables ϵ_θ , $\epsilon_{\dot{x}}$ and $\epsilon_{\dot{y}}$. The speed of the device is assumed to be affected by additive Gaussian noise of covariance Q . This allows for changes of direction.

The system model for a mobile device is given by the prediction function f .

$$state_{k+1} = f(state_k) = \begin{pmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \\ \dot{x}_{k+1} \\ \dot{y}_{k+1} \end{pmatrix} = \begin{pmatrix} x_k + \dot{x}_k \delta t \\ y_k + \dot{y}_k \delta t \\ \theta_k + \epsilon_\theta \\ \dot{x}_k + \epsilon_{\dot{x}} \\ \dot{y}_k + \epsilon_{\dot{y}} \end{pmatrix} \quad (3.6)$$

where δt is the time passed between the previous update of the estimate for this device and the present measurement. The matrix A is the matrix of partial derivatives of this function with respect to the state variables. Note that we do not model the speed of rotation because, in our experimental setup, mobile sensors move in straight lines most of the time and only rotate briefly.

Alternatively, a mobile device's movement can be modeled as random. The state no longer needs to include velocity, and the position is directly affected by Gaussian noise. The prediction function becomes

$$state_{k+1} = f(state_k) = \begin{pmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{pmatrix} = \begin{pmatrix} x_k + \epsilon_x \\ y_k + \epsilon_y \\ \theta_k + \epsilon_\theta \end{pmatrix}. \quad (3.7)$$

3.2.4 Real deployments

In this section, we describe our deployment scenarios, and characterise the nodes' raw range and bearing accuracy. In these experiments, we use the Relate *bricks* already described in section 3.1.1. Figure 3.6 gives a more detailed view of the hardware. These nodes share measurements with each other over the RF channel. For experiment logging purposes, one node reports this data to a computer via a USB connection.

In a 2.75×2.00 m arena, we ran two types of experiment: (1) fifteen static nodes; (2) five static nodes and one mobile node. Mobile nodes were carried by Lego Mindstorms robots, which were programmed to run in a straight line until the border of the arena was hit, and then turn back into the arena and proceed onward. In order to get accurate, real-time groundtruth location/orientation of nodes, we used two cameras suspended above the arena and ReacTIVision software developed by Bencina and Kaltenbrunner (2005) to track visual markers attached to the tops



Figure 3.6: Ultrasonic sensor node: a Relate *brick* — microcontroller and radio transceiver on the top board, transducers and amplification circuit on the lower board.

of all the nodes. In many applications, this camera-based tracking system is not a viable alternative to other localisation modalities because it requires installation and calibration, and only covers a small area. We found that this system gave centimetre-level resolution over most of the arena, with slightly lower accuracy in the corners due to the distortion introduced by the wide-angle camera lenses. Each experiment was run five times for five randomly generated layouts of the nodes, and data was collected for approximately five minutes each time. Our full dataset from both types of experiment contains over two million range and bearing measurements between nodes.

Our first set of experiments, shown in figure 3.7, involved fourteen static nodes and six mobile nodes mounted on Lego Mindstorms robots. The poor results for the mobile nodes in this experiment prompted us to run a second set of experiments with only five static nodes and one mobile node. Using fewer robots increased the effective update rate due to lower contention rates, but not enough to successfully track the mobile node. We also ran a third set of experiments involving fifteen

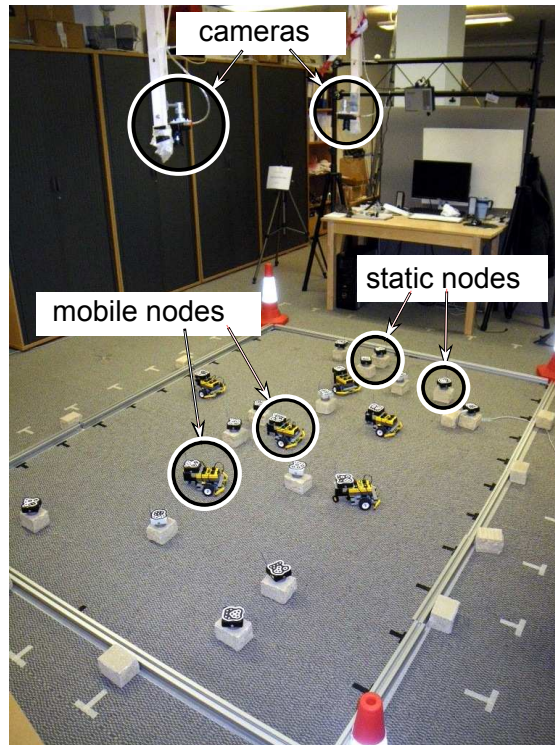


Figure 3.7: Deployment of nodes and mobile nodes mounted on Lego Mindstorms robots. The two cameras suspended above the arena are used to measure the real positions of the nodes. (Due to problems with the measurement rate this particular experiment was not used in our analysis.)

static nodes and no mobile nodes in order to increase the size of our total data set and get a more accurate error distribution for use in simulations.

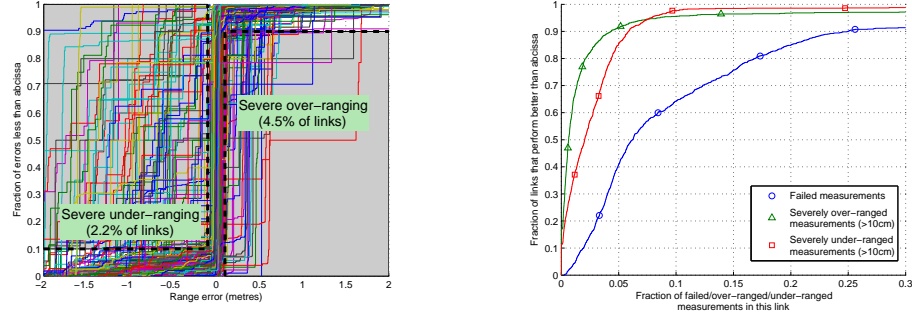
Raw measurement errors. As has been previously observed for other ranging sensor systems (Whitehouse and Culler, 2006), the ranging error of our ultrasonic nodes is not Gaussian. In fact, looking at the range error distributions for single node pairs (fig. 3.8a), it is clear that a significant proportion (about 7%) of links systematically under- or over-range by more than 10 cm. This is similar to Whitehouse and Culler’s proposed “Model 3” error, which accounts for under- and over-estimates. However, in our case, the large offsets for measurements between a given node pair in a particular spatial configuration are repeatable (i.e. the error has a significant systematic component). This systematic error is heavily dependent on the relative location and orientation of the particular node pair.

We illustrate the amount of systematic error which occurs in our experiments in

figure 3.8a. We designate a link as under- or over-ranging if more than ten percent of its measurements were worse than 10 cm. These limits are represented by the dotted black lines. Any node pairs with error distributions outside of those lines have high proportions of systematic error. Figure 3.8b gives further detail on the frequency of over- and under-ranging which occurred on each link. For example, one can read from the plot that 1% of links under-ranged, and 2% of links over-ranged in more than a quarter of their measurements. We additionally show the per-link measurement failure distribution; 35% of links failed to detect an in-flight ultrasonic pulse more than 10% of the time. (As explained below, these failures are due to the ultrasonic sensing; they do not include measurements lost due to radio communication failure.) For the node pairs in which one or both of the nodes is mobile, there were no such systematic offsets.

In total, from our five and fifteen static node experiments, we have over a thousand per-link error distributions. For each link between a static node pair in our simulated evaluations below, we randomly choose a real link error distribution, which is used to model range error, bearing error, and measurement failure for that link throughout a particular simulation.

Packet loss and measurement failure. The physical dimensions of our deployments were such that all nodes should have been within measurement range of each other. Thus, a pulse emitted by a node should cause each of the other nodes to produce a range and bearing estimate. However, we observed that for most emitted ultrasonic pulses, less than half of the other nodes reported a measurement. Fig. 3.9 shows that the measurement rates achieved in our experiments are much lower than the expected values. This low measurement rate can be attributed to two causes: (1) *measurement failure* (characterised above), where the receiving node was unable to detect a valid ranging signal; and (2) *radio packet loss*, affecting the trigger packet sent out by the transmitting node as it emits the pulse, and/or affecting the range/bearing readings reported by receiving nodes to the logging computer.



(a) Range error CDFs, for each static node pair. The thick dotted black lines indicate 10cm error and 10% of measurements. Any node pairs with error distributions outside of those lines have high proportions of systematic error.

(b) Distribution of per link ranging quality.

Figure 3.8: Distribution of per link severe under/over-ranging rate (error greater than 10cm) and measurement failure rate (RF sync packet received but no ultrasound pulse detected). Both plots generated from 1010 one-way node-to-node links.

The maximum aggregate ultrasonic pulse transmission rate⁴ is about 10 pulses per second for our devices. Each transmitted pulse should generate a measurement at each receiver within range. Thus, six nodes (five static and one mobile) should yield an aggregate rate of 50 measurements/s, and fifteen nodes 140 measurements/s. Total losses were about 38% for the six-node case, and 59% for the fifteen-node case. Since nodes always report the ranging result (even when the ultrasonic measurement failed), we can compute the readings lost due to measurement failure (the difference between “all reported measurements” and “valid measurements only” on fig. 3.9).

It is clear that the majority of the measurement losses are due to packet loss. This is typical of the legacy radio module and MAC protocol used on the “brick” devices. In a separate set of experiments, Agbota (2009) found packet losses to be greater than 55% for a single-hop network of twenty nodes. With contemporary radios (CC2420) running the TinyOS MAC in the same setup, he found packet losses to be only about 5%.

⁴The *aggregate* transmission rate is the number of ultrasonic pulses transmitted per second for all nodes. The individual rate is less because the ultrasonic channel is shared between them.

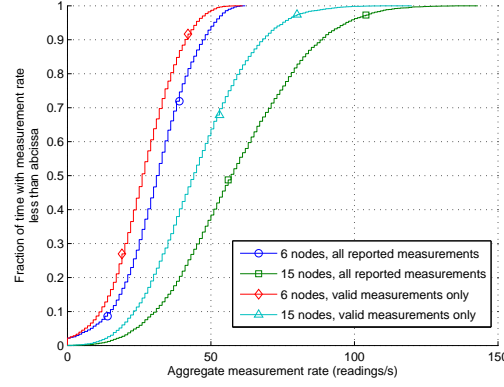


Figure 3.9: Distribution of measurement rates for six nodes (five static and one mobile), and for fifteen nodes (all static).

Low total loss rates are particularly crucial for the dynamic tracking of mobile nodes. In our simulations below, we have neglected radio packet loss, since near-perfect packet delivery to single-hop neighbours is achievable with modern radio modules, especially if one is careful to coordinate transmissions to minimise contention. However, we have taken into account loss due to ultrasonic measurement failure; we even model node pairs with nearly broken links (more than 90% measurement failure), in accordance with the proportion of such links observed in our real data. This implies that dynamic tracking should be achievable with suitable hardware.

3.2.5 Evaluation

We evaluate the accuracy of the Kalman filter for both static and mobile nodes. Using the data from our single-hop deployments, we simulate larger multihop networks. We also simulate the dynamic tracking of nodes, since the total loss rate in our deployments was too high to track our mobile robots. However, for all simulated conditions (multihop and/or mobile), we always draw from the real per-node-pair measurement error distributions.

Computing relative localisation error. Like any relative (anchor-free) positioning algorithm, our Kalman filter produces position and orientation in an arbi-

trary coordinate system that can shift slightly between updates. Thus, to evaluate the accuracy of the estimates, we use the Kalman-estimated positions and orientations to compute the distances and bearings between all pairs of devices after each measurement. We then compare these distances and bearings to those derived from our ground truth computer vision system. In a network of N nodes, there will be $N(N - 1)$ distance and bearing errors in total. These determine the accuracy of the system seen from the point of view of the nodes. Equivalently, this would be the accuracy visible to a user equipped with one of these devices and viewing the spatial layout on a handheld screen.

Locating static devices

Initialisation method and convergence (real data). As mentioned on page 58, we implemented several ways to initialise the Kalman filter. Fig.3.10 compares convergence after initialising the filter with: (1) naive initialisation – a single range/bearing measurement to initialise the position of a node and another single measurement for the orientation; (2) graph trace initialisation – a block of the first $N(N - 1)$ measurements is used to roughly locate all possible nodes using simple trigonometry. Convergence is reached after about fifteen seconds for both methods in the five and fifteen node experiments; this corresponds to approximately five rounds of measurements in the fifteen node case (i.e., each node transmitting a pulse five times in total). In theory, assuming an aggregate pulse transmission rate of 10 pulses per second, five rounds of measurements should only take 2.5 seconds in a five node network, and 7.5 seconds in a 15 node network. However, with our poor packet delivery conditions, it takes at least twice as long to collect the same number of measurements.

Static single-hop performance (real data). We compare the accuracy of our Kalman filter with the accuracy of the raw range and bearing measurements, with a centralised non-linear regression (NLR) algorithm, and with MDS-MAP

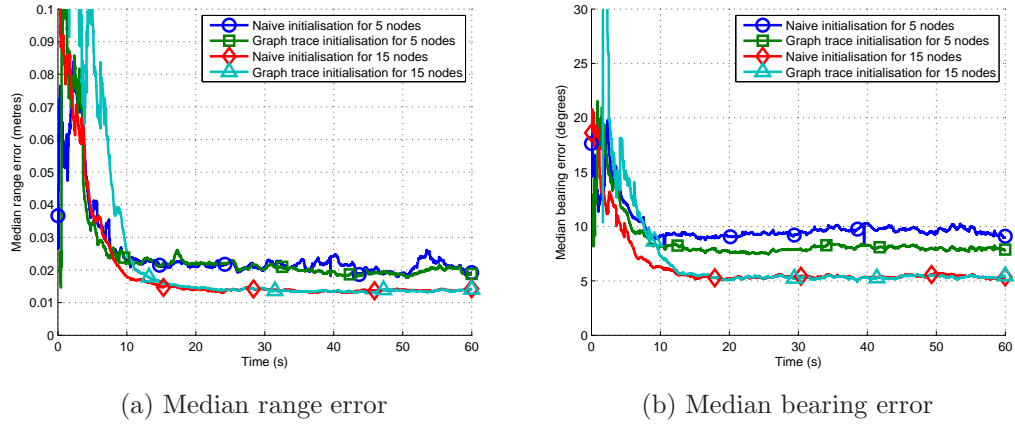


Figure 3.10: Median accuracy convergence of the naive and graph trace initialisation methods averaged over 25 real experiments with 5 and 15 nodes.

(a popular sensor node localisation algorithm by Shang and Ruml (2004)). For each trace, we run the NLR and MDS-MAP on 500 successive blocks of $N(N-1)$ measurements⁵ (where N is the total number of nodes in the experiment) in order to illustrate their localisation error distributions.

Fig. 3.11 also shows that in the static node experiments, 90% of the Kalman relative position estimates are under 4 cm. This is slightly better than the unfiltered range measurements of which 90% have errors under 7 cm, and considerably better than the output from MDS-MAP where more than half the estimates have an error greater than 20 cm. The bearing errors improve considerably; the 90th percentile bearing error is more than halved from 38° in the unfiltered measurements to less than 15°. We remind the reader that although we compare range and bearing errors, these are derived from the Kalman filter’s coordinate and orientation result for each node. Such relative spatial information (in a single coordinate system) isn’t directly available from the raw range and bearing measurements.

We see that our filter is close to the accuracy of NLR for our deployments, which is a good result considering the computational expense of the regression’s large matrix inversions and gradient descent method. Our estimates are also more

⁵These algorithms can produce estimates using fewer measurements, but these need to be selected so that all nodes are connected. In practice, $N(N-1)$ ensures that all nodes are connected and locatable.

stable over time than the NLR or MDS-MAP estimates, although this is only visible in error versus time plots for individual experiments which are not presented here. The additional stability is expected because only our filter takes previous estimates into account and has a built-in smoothing effect.

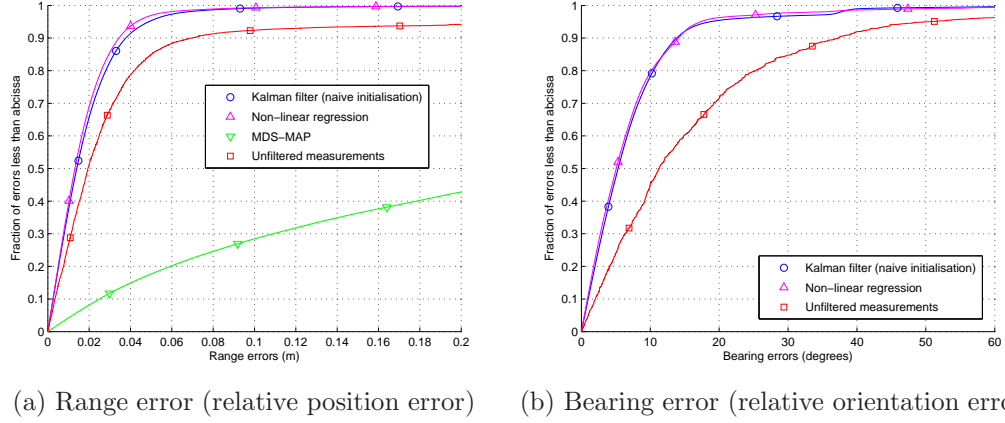


Figure 3.11: Distribution of relative localisation errors for static nodes. The raw measurement errors, the localisation results from a centralised, computationally expensive nonlinear regression algorithm, and the results from the popular MDS-MAP algorithm are shown for comparison. Aggregate data from experiments with five and fifteen static nodes.

Accuracy using ranges only (real data). Because not many sensor platforms are capable of taking both range and bearing measurements, we evaluated the performance of the Kalman relative positioning algorithm operating on the range-only data. For our real deployments, the static accuracy was nearly identical to that of the Kalman filter which used range and bearing (fig. 3.11). The median and ninetieth percentile errors are within 1 mm, and the ninety-ninth percentiles are within 6 mm of each other. Although in our case bearing measurements do not significantly improve the location estimates, they are essential in determining the orientation of the nodes and in resolving “flip ambiguities” (Moore et al., 2004) during initialisation.

Static multihop performance (simulated). Based on the error distributions and measurement failure rates for each node pair in our real deployments, we simulate larger, multihop static deployments which would realistically occur when

the area covered by the sensors extends beyond their measurement range. For each node pair within measurement range of one another, the simulator (a Matlab script) randomly chooses a range/bearing error distribution and measurement failure rate from a real node pair (fig 3.8a). Measurements between the two nodes are drawn from the same distributions throughout that simulated layout, to capture both the systematic error and measurement failure qualities of our nodes. The simulations involve twenty-five nodes on a 15×15 m grid with a 3 m spacing. The maximum sensing range was capped at 5 m (similar to our actual sensors) so each node could only see at most eight neighbouring nodes. In these static simulations, the nodes take turns to transmit a pulse with an aggregate rate of 10 pulses per second (we assume lossless communications). When initialising the Kalman filter using the naive method or the graph trace method, results were poor. These simple methods are not robust enough to provide good initial estimates for multihop networks and the Kalman filter is unable to correct large scale errors. In an attempt to improve this, we tried two alternative initialisation methods. One is similar to the naive initialisation but uses the median of ten measurements instead of a single measurement. The other is MDS-MAP (Shang and Ruml, 2004) which has the benefit of considering the global map, not just local relationships between nodes.

As fig. 3.12 shows, the multihop localisation accuracy with MDS-MAP initialisation is better than 25 cm and 15° after two minutes, for 90% of estimated ranges and bearings. The other initialisation methods all perform poorly for this type of multihop network although taking the median of ten measurements does bring a slight improvement. Due to large, systematic range and bearing errors, the naive or graph tracing methods (which in fact operate on a similar principle) often provide a globally distorted view of the network, even though relative location estimates are correct in many local neighbourhoods. In a multihop network, the Kalman filter is unable to correct this type of errors because it also works locally. However, in this larger scale multihop topology MDS-MAP is able to generate a much more accurate view of the global network by using hop counts in addition to the range

measurements. After using the bearing measurements to detect and correct a potentially mirrored MDS-MAP result (a shortcoming of all anchor-free, range-only methods), the Kalman filter can then refine the location estimates further. The results in fig.3.12 were obtained using this technique.

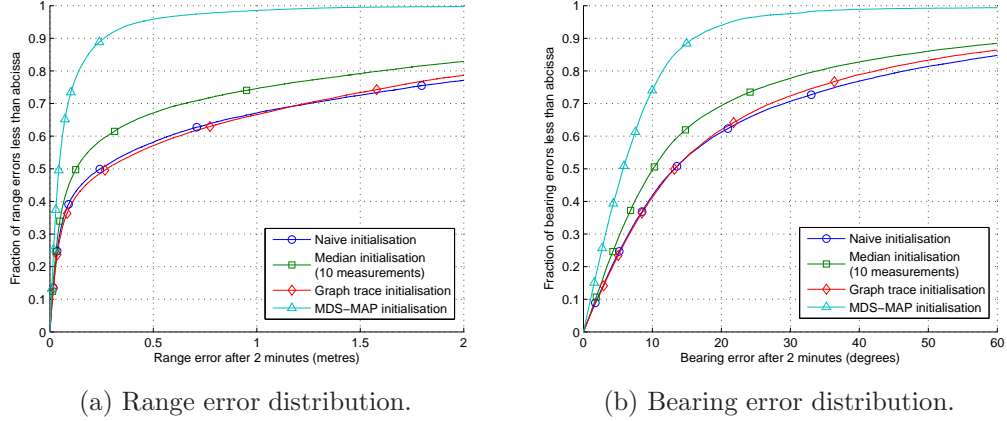


Figure 3.12: Range and bearing error distributions after two minutes in a multihop network. 25 nodes in a 15×15 m grid. Ten pulses per second, aggregate distributions over 100 simulations.

Fig.3.13 shows how long the initialisation takes for each of the initialisation methods described. Again, the naive and graph trace methods perform similarly, but the graph trace method needs to wait about 10 seconds to collect enough measurements. The method which initialises sensors using the median of 10 measurements for each sensor pair performs slightly better but waits even longer, approximately 25 seconds, to collect the required measurements. MDS-MAP initialisation enables the relative Kalman filter to perform best in terms of accuracy and only requires ten seconds to collect enough measurements.

In summary, the particular initialisation method used depends upon the application. For larger, multihop networks where global topological correctness is important, a lightweight algorithm such as MDS-MAP should be initially run on a network-wide (global) batch of measurements. Then, each node's Kalman filter can be used to greatly improve the MDS estimates and accurately track mobile nodes in real-time (as shown below). For other situations, where only local neighbour-

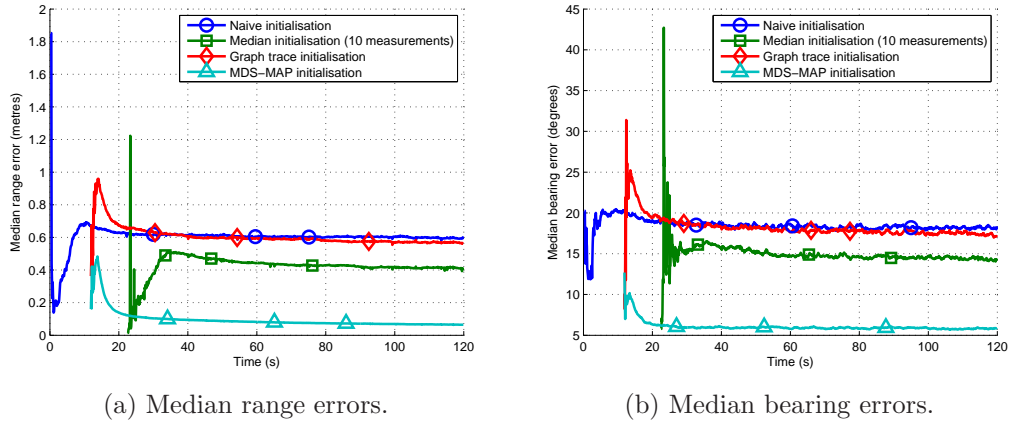


Figure 3.13: Multihop initialisation performance. Twenty-five nodes in a 15×15 m grid. Ten pulses per second, average errors over 100 simulations. Compare to single-hop initialisation in fig. 3.10.

hood correctness is important, it may be sufficient to use a simpler initialisation method for the Kalman filter running on each node.

Tracking mobile devices

As mentioned in section 3.2.4, we were unable to directly perform real-time tracking on the mobile nodes in our gathered data traces because the measurement loss rate was too high. Thus, to evaluate the real-time tracking performance of the Kalman filter, we rely on simulations. Raw measurements involving our mobile devices have slightly worse error characteristics (about 2 cm worse for ranging, at the ninetieth percentile) than measurements between static devices, but large error offsets tend not to be recurring. This is because the mobile devices do not stay for long in a position that systematically produces over- or under-estimates for range and bearing. Thus, in our simulations, we sample the mobile node measurement errors from an aggregate error distribution of the real-world data gathered to and from mobile devices, where the measurement failure rate is 15% according to that same data. Measurements between static devices are generated as described earlier. As before, all devices take it in turns to transmit an ultrasound pulse which is received by all other devices (subject to the measurement failure rate for each particular

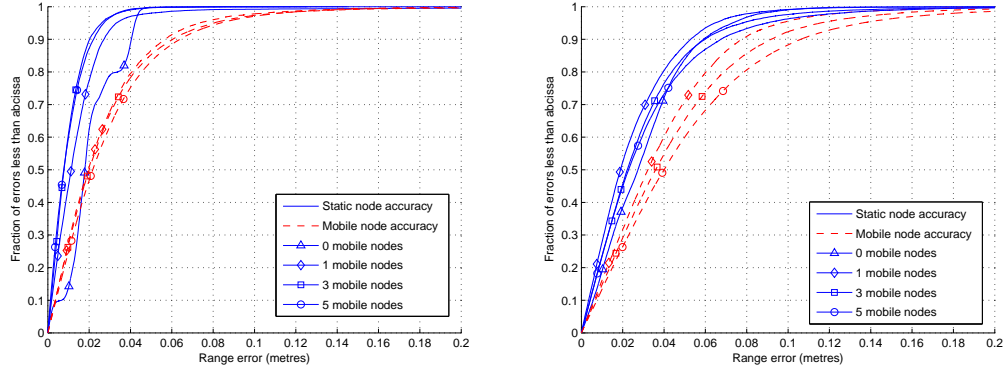
link). Thus, for every pulse, up to $N - 1$ range/bearing measurements are produced. We vary the rate at which pulses are emitted from 10 to 50 per second (aggregate pulse rate).

In this simulation, most nodes are within measurement range of one another, most of the time. There are five static nodes placed randomly in a 5×5 m arena, wherein varying numbers of mobile device move. Like the Mindstorms robots in the real deployments, a simulated mobile node travels in a straight line at a constant speed until it reaches an edge, then it changes direction to move back into the arena.

Movement models and real-time tracking accuracy. Fig. 3.14a illustrates the tracking accuracy, where the Kalman filter running on the mobile nodes uses a constant velocity model (eq. 3.6). Note that tracking accuracy does not significantly worsen as the number of mobile nodes increases from one to five. In the case where there are five mobile nodes, 50% of the network is mobile. The assumption that the nodes travel at constant velocity causes the position estimates to overshoot the edge of the arena due to the sudden change of direction but this is quickly corrected especially with higher measurement rates. We were able to observe this in a number of videos⁶ generated from a sequence of estimated positions in Matlab. For each frame, we use a Levenberg-Marquardt minimisation to compute the optimal transformation (2-dimensional rotation and translation) mapping the estimated node coordinates to the real node coordinates, before assembling them into a continuous animation. In a dynamic scenario, these animations help us visualise and evaluate the quality of the estimated coordinates more effectively than a static graph.

However, for some applications it may be more desirable to apply the same, generalised model to all nodes, and not require that they have knowledge of whether they are static or mobile. Fig. 3.14b shows the accuracy when all nodes (static and

⁶<http://www.youtube.com/user/kalmanvids/videos/> (Accessed 2012.09.24.)



(a) Constant velocity model for mobile nodes (b) Random movement modelling for static and mobile nodes

Figure 3.14: Kalman filter dynamic tracking accuracy. Five static nodes. Speed of mobile nodes = 1 m/s, aggregate pulse rate = 50 pulses/s. Averages over 20 simulations.

mobile) run the same filter. They are all allowed to move in any direction, only constrained by their range and bearing measurements (eq. 3.7). As might be expected, accuracy for both static and mobile nodes worsens. Moreover, the dynamic tracking accuracy visibly worsens as the number of mobile nodes increases—from 8 cm for one mobile node, to nearly 11 cm for five mobile nodes (90% confidence).

Fig. 3.15 illustrates the effectiveness of the different movement models, side-by-side. Clearly, applying a random movement model to the mobile nodes worsens their tracking result, which is not surprising for the type of movement in the simulations. Allowing static nodes to move freely worsens their result (3 cm to about 5 cm, 90% confidence). However, this general measurement model can deal with the case where all nodes are mobile. This is in contrast to other techniques which require a large proportion of the network to be static, or the mobile nodes to follow a particular type of trajectory. We have not plotted the accuracy of our Kalman filter in scenarios where all nodes are mobile, but the real-time simulated animations (see p.76) show promising results.

Measurement rates for effective tracking. To quantify the effect of speed on the tracking accuracy for the mobile device we simulate a mobile device moving at different speeds in networks having different measurement rates. As one

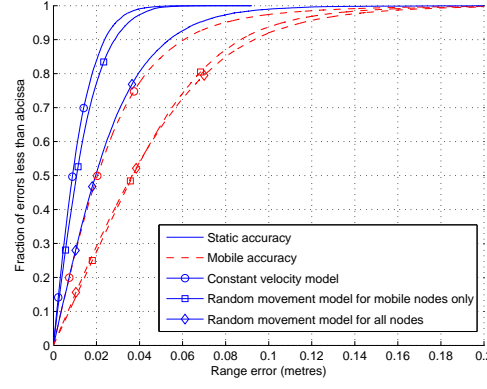


Figure 3.15: Comparison of tracking accuracy for different Kalman movement models. Five static nodes and three mobile nodes. Speed of mobile nodes = 1 m/s, aggregate pulse rate = 50 pulses/s. Averages over 20 simulations.

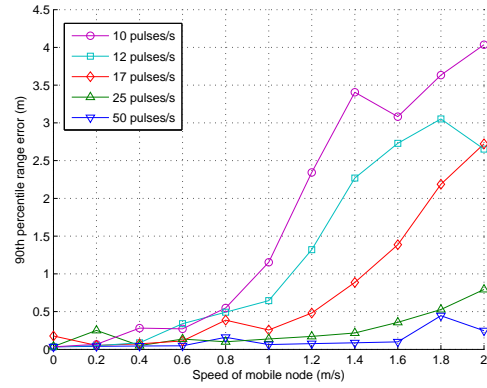


Figure 3.16: Mobile tracking accuracy for different aggregate pulse emission rates. The 90th percentile errors are averaged over ten simulations of five static devices and one mobile device in a 5×5 m arena.

would expect, devices moving at higher speeds require higher update rates to be tracked with the same degree of accuracy (fig. 3.16). For instance in order to track devices moving at a walking speed of about two metres per second with an accuracy greater than fifty centimetres, the aggregate pulse rate needs to be between 25 and 50 pulses per second. We recognise that the accuracy is also dependent on the number and geometric configuration of the static nodes. Establishing the effect of these parameters requires further work.

3.2.6 Suitability of the relative Kalman filter for wireless sensor network localisation

Although we have shown that our proposed algorithm works with real data, and have addressed some of the limitations of our experiments through additional realistic simulations, there are a number of questions that remain to be answered in practice. Nonetheless, our results give us some insight into how this algorithm would perform in a large scale experiment using dozens of nodes spread over tens of metres, with our Kalman filter implemented on the sensor nodes themselves.

Distributed computation on sensor nodes

Accurate and sophisticated localisation algorithms can be prohibitively expensive for sensor nodes. Although we developed and tested the relative location Kalman filter on a workstation PC, it is suited to be distributed in a sensor network. Each device would store its own location and error covariance. The RF trigger packet that precedes the ultrasound pulse would include this information, thus enabling all receivers to run the filter for this measurement and update their own location. Only information about the two devices involved in a measurement is required to process that measurement, and each measurement only affects the location estimates of those two devices.

In figure 3.17, we show the floating point multiplications required of each node running the Kalman filter, assuming it is able to take range/bearing measurements to each of its neighbours. The Kalman operations are computed based on those tabulated by Groves (2008, tab.3.1). We also show the per-node computational cost if each node produces updates based on five measurements to each of its neighbours, to reflect the amount of computation needed for convergence in our experiments. This is equivalent to the number of measurements taken during the

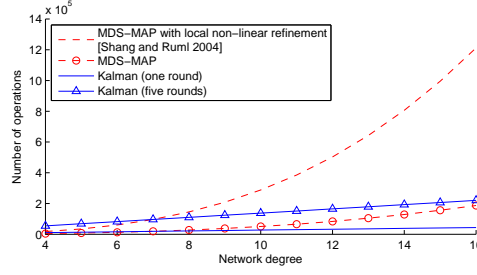


Figure 3.17: Per-node computational complexity of the relative Kalman filter (floating point multiplications).

first 15 s for the 15 node deployments in figure 3.10. The per-node computational cost of MDS-MAP (Bischoff et al., 2006, tab. 1, neglecting the stitching and global refinement stages), is plotted for comparison. (Of course, MDS-MAP) is not well-suited to dynamic tracking, while the Kalman filter is.) For networks of small degree, the computational cost of relative Kalman filtering is comparable to MDS-MAP, and as the number of neighbours rises above seven, Kalman filtering becomes much cheaper. MDS-MAP with no refinement is computationally comparable to Kalman filtering, but it has an accuracy far worse (fig. 3.11a). MDS-MAP is, however, a good candidate for a one-time initialisation of the Kalman filter, if the network has many hops and global topology correctness is important.

Note that the batch-based algorithm employed by Taylor et al. (2006) is also significantly more expensive than Kalman filtering. Because multiple static node positions, and one or more mobile node positions (depending on the number of events in the batch) must be estimated, the matrix inversions required are significantly larger than the simple two-node solution computed in each of our Kalman updates. It is cheaper to perform several two-node updates than a single many-node update.

Communication overheads

All distributed algorithms incur some communication overhead as the local map is computed. For our algorithm, each receiving node needs to share its location

and orientation updates (pertaining to itself and the transmitting node) with its one-hop neighbours. The neighbours use these location and orientation results to update the current state of their filters. As others have proposed, these updates could be piggybacked on other network traffic (notably the packets used to announce the emission of a ranging signal). Or, to allow updates to be delivered with as little latency as possible, guaranteed, periodic time slots could be assigned to nodes within two-hop neighbourhoods (Rhee et al., 2008).

Requirements for effective localisation of mobile devices

Tracking mobile devices requires a high measurement rate but because of the slow physical speed of ultrasound (approximately 340 m/s), the measurement rate is limited by the maximum range. For instance if we want a maximum range of 5 m then the devices must wait at least 15 ms after each pulse before another can be sent. This is equivalent to a maximum aggregate pulse rate of 67 pulses per second. The measurement rates reported in this paper may appear higher than those required by alternative algorithms for several reasons. First, we make no assumptions on the type of movement and have no direct measurements of the speed of the mobile nodes. A higher measurement rate is therefore required in order to acquire more information about each node. Second, all nodes are transmitting ultrasonic pulses in round-robin fashion. Pulses transmitted by a mobile node generate useful measurements at each of the receivers, but pulses transmitted by a static node with a known position only generate a single measurement which is helpful in locating the mobile node, the other measurements between static nodes are wasted. In the typical case of a single mobile node travelling among static nodes, only the mobile node would need to transmit pulses and the required measurement rate would be lower. The measurement rates required by our particular sensor network may not be appropriate for some applications, but could be decreased if different sensing modalities are used or if additional constraints concerning the movement of the nodes are available.

Variations on the filter

Our filter could be modified to process all measurements that were generated by the same pulse simultaneously rather than one after the other. The state vectors and covariance matrices would have to contain values for all nodes instead of just two but the equations would be otherwise unchanged. According to Simon (2006, p.150), both methods are equivalent. We prefer the sequential formulation where measurements are processed one after the other because it requires smaller matrix inversions.

Our approach could also be adapted to use a particle filter (Arulampalam et al., 2002) instead of the Kalman filter. The particle filter has the advantage of being able to model non-linear non-Gaussian systems such as ours. It also has the benefit of dealing well with the multimodal position distributions which we observed during the initialisation phase. However, it typically requires hundreds or thousands of particles to model the position of each node and each of these particles must be re-evaluated for each measurement. This requires orders of magnitude more computation than the Kalman filter which only needs to re-evaluate a single mean and covariance for each measurement. In principle, it also makes the algorithm a lot more difficult to distribute over devices because there is so much information to share. However, Challa et al. (2002) describe how support vector machines can be used to compress the particles before transmitting over the network, and Liu et al. (2009) successfully use this method to track a single mobile target in a field of static sensor nodes in a simulation.

Generalisation

In this section, we have used a specific type of sensor node using a specific sensing modality and a specific communication scheme. However, our method and conclusions can be generalised to many kinds of sensor network which require localisation.

The Kalman filter we have described is easy to adapt, even if the range and optional bearing measurements have different characteristics from ours, or if an altogether different type of measurement, such as speed or signal strength, is available. Equally, the measurement scheme is not limited to the simple round-robin scheme where all nodes take turns at transmitting a signal. For example, in the scenario where a mobile node moves through a field of stationary nodes, it may be best if the mobile node constantly sends ranging pulses, in order to optimise tracking accuracy and latency (since the measurement rate will be much higher than a simple round robin scheme might allow). Different applications will have different requirements, but will also be able to provide additional constraints for the localisation problem which can be incorporated into the filter, improving performance. Our point is that the Kalman filter is a low-overhead, scalable and accurate algorithmic tool for localisation in sensor networks, and can be adapted to a variety of measurement modalities and protocols, node mobilities, node densities, and network sizes.

Benefits of the relative Kalman filter approach

The Kalman filter is an untapped resource for wireless sensor networks and is able to do more than just track a single mobile node among a network of static anchor nodes. It is a competitive localisation solution in terms of accuracy, configurability and adaptability, computational requirements, and communication overhead. We have illustrated this by characterising a Kalman filter that is able to compute the real-time location and orientation of sensors in an arbitrary reference frame. The filter works with range and bearing measurements and has been shown to perform as accurately as a centralised non-linear regression algorithm on single-hop deployments. In simulated deployments over larger areas, with several hops between devices, our filter converges well, provided it is initialised with a roughly correct global topology, such as that produced by MDS-MAP. We have also shown that the filter can accurately track multiple mobile devices as long as the effective

measurement rate is high enough with regard to the speed. Real-time tracking is even possible in the case where all devices are mobile. Not only is the Kalman filter adept at accurate, real-time tracking of mobile nodes, but it also accurately positions static nodes, at a computational overhead which scales linearly with the local neighbourhood size.

A similar type of Kalman filter is used in chapters 5 and 6 to perform simultaneous localisation and mapping for pedestrians. The main difference is that, in those later chapters, we are able to measure the movements of the mobile node using inertial sensors with the methods explained in chapter 4. This gives us more information about its trajectory and helps us initialise the positions of the stationary sensor nodes.

3.3 Discussion on the merits of ultrasound for localisation

Ultrasound is an appealing way to perform localisation, especially on low power and low cost sensor nodes. The relatively slow speed of ultrasound means we only need nodes with basic microcontrollers and simple electronics. Calibration need not be painstakingly accurate because many of the timing errors introduced by the radio transmission or processing will only translate to small errors in distance. Ultrasound is easily blocked by obstacles; this could be considered a disadvantage because it reduces connectivity between nodes and thus the information available for estimating locations, but it also works to our advantage by reducing interference between nodes in different rooms and allowing us to acquire topological information rather than purely geometrical (e.g., two nodes that are on opposite sides of a wall will not be able to see each other, although they are geometrically close). This is useful in the firefighter navigation scenario.

Despite these advantages, we also observed several limitations of ultrasound technology which should be taken into account when designing future experiments or systems. Below, we describe several avenues of research which should help improve these points.

- Ultrasound is relatively slow and this inherently limits the measurement rate.
- Individual measurements, particularly bearings, are unreliable in realistic settings. This is even more pronounced for mobile devices.
- The measurement scheme is important to reduce the effects of the two previous points.
- Our sensor nodes were designed to work in two dimensions, with all devices in the same plane (e.g., on a tabletop or on the floor), not in three dimensional space.

As mentioned earlier, the slow propagation speed of sound waves has some advantages. It allows us to use low-cost hardware to measure time-of-flight and still achieve errors of only a few centimetres. This is in contrast with the sophisticated hardware and software algorithms required for radio time-of-flight or time-difference-of-arrival measurements. However, working with sound means we must allow a considerable delay for them to travel between the transmitting and receiving nodes. This delay is proportional to the maximum distance we expect to measure between the devices. Our first ultrasound prototypes waited approximately ten milliseconds and thus had a maximum range of 3.5 metres. Later, we increased this to five metres, but each measurement takes an additional four milliseconds. Due to the limited resources of our sensor nodes, receivers can only listen on a single transducer at a time. Each transducer only covers approximately a 90° sector. Therefore, the transmitter sends the ultrasound pulse on all transducers four times. Each time, the receiver listens on a different transducer. This means that

each measurement effectively takes four times longer, and a pair of sensor nodes can take at most seventeen range and bearing measurements per second. If we allow enough time to measure ranges up to ten metres (and if the amplitude of our pulses is strong enough) then we can only take 8 measurements per second. If more than one device needs to transmit, it will need to wait its turn, thus reducing its own measurement rate even more. This is in addition to the delays occurring in the radio stack, due to channel management, collisions and packet loss. Once again, more advanced hardware would allow us to sample all four transducers simultaneously and save time, and to benefit from the information contained in the phase of the pulse to refine both range and bearing measurements. The pulses could also be frequency-multiplexed or pipelined to increase the effective measurement rate. With some additional work on the hardware and low level measurement code, achieving a pulse rate of 50 pulses per second (as required for tracking mobile devices) is realistic.

Ultrasound measurements are sensitive to the environment. Two stationary devices may measure range and bearing with systematic error due to reflections. These systematic errors are in addition to the random errors, typically modelled as Gaussian, due to air movement, acoustic noise or electronic noise. When one of the devices is moving, the errors are unlikely to be systematic but the measured values generally do not vary smoothly and include many outliers because each of the four successive pulses is in fact sent from a slightly different position. This is a limitation of the specific hardware we used. Individual measurements are not suitable for many applications due to these systematic errors and numerous outliers. A moving average filter is not sufficient to correct these problems but more sophisticated filters such as the relative Kalman filter we have described above can produce reasonable estimates even with poor measurements.

In our particular system, only the receiver can estimate the direction towards the transmitter. This information relates the position and orientation of the receiver to the position of the transmitter, but the orientation of the transmitter

is undetermined. This makes it important to think carefully about which devices transmit and which receive ultrasonic pulses, depending on the application. If there is a single mobile device moving among many stationary devices spread over a wide area, it may be necessary to have only the mobile device transmit to ensure as many measurements as possible from it. However, if the application needs to know which direction the mobile device is facing, then the stationary devices will also need to transmit. In this case, it would be preferable to design a transmission scheme where only those devices closest to the mobile device transmit, in order to save time and to ensure the system remains scalable. In our simulated experiments with the mobile sensor nodes, the measurement rate was acceptable because all devices were within range of each other, therefore each measurement was used to estimate the position of all the nodes.

Our relative Kalman filter is only able to track mobile nodes if the measurement rate is high enough. In the area of wireless sensor networks, this is an issue because the communication and measurement overhead reduces battery life. In our application to emergency response, however, the system only need work reliably for a few hours at most, rather than the weeks or months that some WSN research tries to achieve. Thus, the requirement of a high measurement rate is not necessarily a problem for our research.

The sensors themselves are designed to work in a plane. The three or four transducers transmit and receive best within a plane due to their directional nature. The localisation algorithms we used all make the same assumption. The system does not fail completely if the planar assumption is broken, but results become even less reliable. A system designed to track sensors in three dimensional space would require different hardware and a modified algorithm.

A number of questions remain unanswered by our work to date and should be investigated through future research. It is not clear how ultrasound range and bearing measurements are affected by the environment at a fire scene. On the one

hand, it is likely that smoke, water vapour and drastic changes in temperature will affect the propagation of ultrasound and maybe of radio waves as well. Noise from the fire itself and from all the human activity at the scene may contain ultrasound frequencies which interfere with our location system, but it may be possible to modulate our signals in a particular way to minimise this interference. On the other hand, high temperatures will not necessarily preclude the use of electronic sensor nodes in this context, because if firefighters are still exploring a building, the temperatures at floor level will not yet be high enough to damage the electronics. Our relative Kalman filter algorithm was developed and tested in the context of wireless sensor networks and a mesh of sensor nodes. In a deployment by an emergency response team, the nodes are more likely to be deployed in a trail. This different type of topology may affect the reliability of the tracking algorithm. Further theoretical models and simulations, as are common in WSN research, would provide some insight into the effect of a trail topology on the tracking ability of our algorithm. However, we believe a real scale experiment, with dozens of redesigned and improved sensors deployed as a person walks through several rooms of a building, is the only way to determine whether this technology and algorithms similar to the one we have described are capable of providing a robust localisation solution.

3.4 Conclusion

In this chapter, we studied two localisation techniques using ultrasound sensor nodes, suitable for use in an uninstrumented environment. We first demonstrated how we use ultrasound range and bearing sensing to guide pedestrians towards certain tagged locations in an otherwise uninstrumented environment. Then we presented an algorithm capable of tracking several of these sensor nodes relative to each other in a fully mobile wireless sensor network.

There are several reasons why these techniques have potential for providing navigation support to emergency responders:

- The systems we present work by design in uninstrumented environments.
- Ultrasound sensors can be built with modest resources but can still achieve the required measurement rates for tracking pedestrians.
- Ultrasound is inherently constrained by walls; this is useful for determining the layout of a building and which areas provide suitable paths for pedestrians.

There are, however, several aspects which require further investigation before focusing on the methods we describe as fully viable solutions:

- Ultrasound range and bearing measurements must be more robust.
- The harsh environmental conditions at a fire scene may affect ultrasound measurements; this must be tested.
- The *trail* topology of the sensor network deployed by the responders may require a different tracking algorithm.

After addressing the points above, researchers should design a full-scale demonstrator to uncover other implementation challenges specific to this application.

We showed that locating a specific item in the environment relative to a person is useful for providing guidance to that person, even in the absence of localisation infrastructure, maps, or anchor nodes with known absolute positions. We also showed that we can achieve this in practice for multiple moving sensor nodes by estimating their positions relative to each other. This is a step towards localisation for emergency responders in unknown environments because it demonstrates that we can provide navigation support without maps or a global positioning system. The sensor node system we used for our experiments was either pre-deployed or simulated, and algorithm initialisation was sometimes problematic, which prevents

this solution from being immediately useful for pedestrian tracking. The following chapter begins to address these concerns by introducing pedestrian dead-reckoning (PDR) which can, among other uses, serve to bootstrap position estimates of nodes as they are deployed.

Pedestrian dead reckoning¹

In our survey of navigation support systems for emergency response (page 23), we explained that inertial pedestrian dead reckoning (PDR) is a method with several key advantages. First, it is completely self-contained, depending solely on the inertial sensors worn by the pedestrian, and not on any external devices which are out of their immediate control. Second, it is, in principle, suitable for any type of motion, whether walking, running, crawling, ascending or descending a ladder or stairs. This autonomy and versatility offer great potential when it comes to our concern for tracking pedestrians in unknown and uninstrumented indoor environments.

Despite the increasing number of researchers who use this technique, some implementations are unnecessarily restrictive and very few authors give a complete description. Although the principles involved are straightforward, our experience has shown that PDR can be unpredictable, especially when it comes to tuning parameters. This chapter provides a reference implementation for foot-mounted inertial PDR and describes many of the difficulties we have encountered.

¹This chapter is a revised version of: Carl Fischer, Poorna Talkad Sukumar, and Mike Hazas. Tutorial: implementation of a pedestrian tracker using foot-mounted inertial sensors. *IEEE Pervasive Computing*, 2013. Accepted for publication. The author acknowledges that Poorna Talkad Sukumar contributed to the development of the PDR algorithm under the author's supervision.

4.1 Introduction

Pedestrian dead-reckoning (PDR) using foot-mounted inertial measurement units (IMUs) is the basis for many indoor localisation techniques which include map matching, various types of simultaneous localisation and mapping (SLAM), and integration with GPS or other localisation techniques. Despite the increasing popularity of PDR methods over the past decade, there is little information about their implementation and the challenges encountered even for very basic systems. Some publications focus on the algorithmic details of PDR and use abstract formalisms which can be daunting to readers who only require a simple implementation. Other publications assume that readers are familiar with PDR and focus on the additional sensors that distinguish their localisation system from others. With this chapter, we aim to make it easier for others to use PDR as a component of a larger system. We contribute in several ways. We provide researchers with a description of a standard inertial PDR method which is simple to implement, is usable with minimal custom configuration, yet represents state of the art in terms of pedestrian inertial tracking. We reference key work which can be consulted for a more formal understanding of the underlying principles, and other works which are dedicated studies of particular aspects of pedestrian inertial tracking. We give an honest account of the difficulties encountered when implementing, using, and evaluating a PDR system.

Our focus in this chapter is on the implementation of inertial navigation systems where gyroscope and accelerometer readings from foot-mounted sensors are integrated to estimate the orientation, velocity and position of the pedestrian. There is a related area of research which counts steps based on the pattern of accelerations, and multiplies them by an estimated step length. This technique can be implemented with inertial sensors worn elsewhere on the body, in more convenient locations such as the waist (Jahn et al., 2010; Goyal et al., 2011). In particular, it is

suitable for implementation on a phone carried in a pocket (Gusenbauer et al., 2010; Jin et al., 2011). Despite this important benefit, in this thesis, we choose to develop our selected method due to its higher accuracy and ability to track even when the pedestrian is not walking forward in a typical fashion. Both methods, however, are self-contained and present similar opportunities in terms of integration with other sensors.

4.2 Key references

Those who want a better understanding of our tracking system and would like to develop it further should become familiar with the area of Kalman filtering, non-pedestrian inertial navigation, and inertial pedestrian dead-reckoning. For their benefit, we give a few key references. Some of the technical terms will be explained later in the chapter.

4.2.1 Inertial navigation and Kalman filtering

Simon (2001) provides a good starting point for understanding the Kalman filter, using a simple vehicle tracking system as an example. Welch and Bishop (2006) are only slightly more formal and give the fundamental Kalman *update* and *predict* equations. They use a simple example to illustrate the effects of adjusting the different filter parameters.

Titterton and Weston (2004) provide a textbook which lays down the foundations of modern inertial navigation. It rigourously defines the different reference frames, and shows how Euler angles, rotation matrices, and especially quaternions can be used to represent the attitude (or orientation) of an inertial sensor. This book explains how an error-state Kalman filter, similar to the one we use, can fuse inertial navigation estimates with estimates from other navigation systems.

Another book on the topic of integrated navigation systems is written by Groves (2008). It includes several chapters covering inertial navigation and Kalman filtering, and includes additional topics such as filter behaviour and parameter tuning. These books provide two complementary views on a complex topic.

4.2.2 Pedestrian dead-reckoning (PDR) using shoe-mounted inertial sensors

Many research papers since 2005 cover the topic of pedestrian inertial navigation. We have selected a few which we found helpful in designing the implementation suggested in this chapter. Ojeda and Borenstein (2007) describe a system similar to our “naïve implementation”. They designed it with emergency responders in mind, and tested it for a variety of walking patterns, on stairs and on rugged terrain. Their simple algorithm performs well thanks to the high-quality IMU they use, which is larger, heavier and much more expensive than our MEMS² sensors. Feliz et al. (2009) describe a similar system but with more emphasis on stance phase detection and velocity error correction. Their system corrects position as well as velocity during zero-velocity updates (ZUPTs), using a less powerful but more intuitive alternative to the Kalman filter we describe. Foxlin (2005) is probably the most cited author in this area. He explains clearly the benefits of using a Kalman filter to apply ZUPTs. Foxlin’s article includes details of the implementation, but in a more recent article, Jiménez et al. (2010) give a more complete description of the implementation process. Their article also gives some tips for tuning a Kalman filter in the specific context of pedestrian inertial navigation. In our opinion, however, these two articles omit some of the essential details required for actually implementing the algorithms they use.

²MEMS stands for microelectromechanical system.

4.3 Inertial pedestrian dead-reckoning

Dead-reckoning is the process of estimating the position of an object by keeping track of its movements relative to a known starting point. A typical example is a ship travelling at a constant speed in a fixed direction. The current position of the ship is on a line starting at the known starting point in the direction of travel, and the distance from the starting point is given by the speed along the direction of travel (i.e., the velocity) multiplied by the time since the ship was at that known point. If the ship changes course, or if the speed changes, the navigator must make a note of the current position estimate and start the process again using this new position estimate as the starting point for future estimates. Over time, these estimates become less and less accurate because they rely on previous estimates, which are imperfect due to errors in speed and heading measurements. In other words, the small errors in heading and speed accumulate to form an increasingly large error in the position estimate.

A basic pedestrian dead-reckoning method could apply exactly the same principles to estimate step-by-step positions, using a (digital) compass to measure the heading and an (electronic) pedometer to count steps. This method works in principle but relies on the assumption that the pedestrian is walking with steps of constant length. Cheap and small microelectromechanical (MEMS) accelerometers and gyroscopes have provided researchers with alternative methods. Typically, these are combined into an inertial measurement unit (IMU) consisting of three accelerometers and three gyroscopes aligned along three orthogonal axes. The accelerations are integrated to estimate velocity and position. Most of the complexity and the error of this method come from the fact that the accelerations are measured in the coordinate space attached to the IMU, the *sensor frame*, and not in a coordinate space easily associated with the room in which the experiment is taking place, the *navigation frame*. This is called a *strapdown* inertial navigation

system because the accelerometers rotate with the object being tracked. This is in contrast to a gimbaled or stabilised platform system where the sensors rotate independently in order to maintain a fixed orientation. The two coordinate systems are illustrated in figure 4.1.

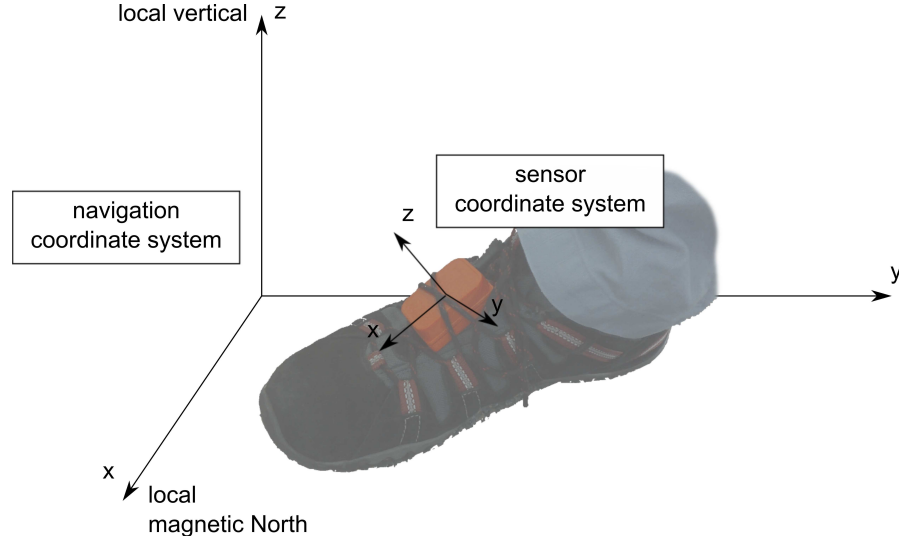


Figure 4.1: Transformation from sensor to navigation coordinates via the direction cosine matrix: $\mathbf{pos}_{\text{nav}} = C_{\text{sensor} \rightarrow \text{nav}} \mathbf{pos}_{\text{sensor}}$.

4.3.1 Attitude and heading reference systems

Many off-the-shelf IMUs include an attitude and heading reference system (AHRS) which estimates the transformation from the sensor frame to the navigation frame (e.g., XSens' MTx³, Intersense's InertiaCube⁴, x-io's x-IMU⁵). In other words, the AHRS computes the orientation of the sensor in 3D space. It outputs the orientation of the sensor as three Euler angles (roll, pitch and yaw), or a 3×3 rotation matrix, or a quaternion. These are all equivalent ways of representing an orientation. We use the rotation matrix notation, which we find the most intuitive.

An AHRS usually combines gyroscope readings with accelerometer, and sometimes magnetometer readings. The integrated gyroscope rates-of-turn give the ori-

³<http://www.xsens.com/en/general/mtx> (Accessed 2012.09.24.)

⁴<http://www.intersense.com/pages/18/59/> (Accessed 2012.09.24.)

⁵<http://www.x-io.co.uk/node/9> (Accessed 2012.09.24.)

entation; the accelerometers correct the tilt of the sensor (roll and pitch), and the magnetometer corrects the heading of the sensor (yaw). These corrections are necessary because the orientation estimated from integrating the rates-of-turn accumulates error from the gyroscope noise. In contrast, the accelerometer and magnetometer estimates of the orientation do not accumulate error, but are affected by the movement of the sensor and magnetic interference respectively. By combining these three types of sensors, an AHRS can provide a good estimate of the orientation of the sensor at all times.

The most straightforward implementations of inertial tracking use the orientations computed by an AHRS. However, many AHRS use proprietary algorithms and are designed to work well for a range of applications. Few are designed specifically for foot-mounted pedestrian inertial tracking where rates-of-turn and accelerations are much higher than those encountered in other fields. We found that by calculating the orientation of the sensor in the inertial tracking algorithm itself, we can track the pedestrian more accurately, and have more flexibility in tuning the parameters of the algorithm. We choose not to use magnetometer readings because, in our experience, they are unpredictable when used indoors due to their sensitivity to metallic objects and interfering magnetic fields in the environment.

4.3.2 Approximations and assumptions

For pedestrian tracking, we can simplify the general inertial navigation equations derived from the laws of physics. There are two reasons for this. First, distances and speeds are much smaller than for aircraft, ships, or land vehicles, thus reducing the magnitude of some of the terms. Second, MEMS inertial sensors have relatively poor error characteristics when compared to the navigation grade sensors typically used for vehicles, and the effect of these measurement errors is much greater than the errors introduced by our approximations.

The full navigation equations compensate for a number of physical effects which we are able to neglect with little consequence for the overall tracking error. For typical walking speeds, neglecting the centrifugal force due to the rotation of the Earth causes a position error of 0.5% of the total distance travelled, if the experiment takes place at the equator, and less elsewhere. Neglecting this effect causes the estimated position to drift down and away from the equator by a small amount. The Coriolis force, the effect by which the rotation of the Earth appears to deflect moving objects, is proportional to the speed of the target relative to the Earth and to the Earth's rate of rotation. For a pedestrian, the effect is several orders of magnitude less than that caused by centrifugal force. In addition, some of the Coriolis errors will cancel out when the pedestrian changes direction, so there is not always accumulation of position error. The Earth's rotation is included in the gyroscope measurements, but the rate of rotation of the Earth ($0.004^\circ/s$) is far less than the bias drift (slow but unpredictable error) of current MEMS gyroscopes (typically $0.1^\circ/s$), and can thus also be neglected. If the tracked pedestrian remains within a few kilometres of their starting point, then we can assume that the curvature of the Earth over this area is negligible; this allows us to work in a traditional Cartesian coordinate system, rather than mapping onto an ellipsoidal surface which approximates that of the Earth.

If we wanted to compensate for all these errors, we would need an accurate estimate of the pedestrian's position and orientation relative to the Earth. This is not possible using only inertial sensors, but requires additional technologies such as GPS, and we have established in earlier chapters that GPS is not a viable solution in many cases, notably indoors.

4.3.3 Zero-velocity detection

Zero-velocity updates (ZUPTs) are an essential part of an inertial tracking system. Applying a ZUPT simply means resetting the estimated velocity to zero.

Without them, the velocity estimate error increases linearly with time and the position estimate error increases at least quadratically (more if we take into account the increasing error in the orientation estimates). The system can apply a ZUPT whenever the sensor is stationary, but detecting the appropriate moment can be challenging. During normal walking, zero-velocity (ZV) occurs during the *stance phase*, when the foot touches the ground. This has made foot-mounted IMUs a popular choice for pedestrian tracking. Tracking algorithms can use sensors mounted on other parts of the body to perform PDR by counting steps and estimating their length, but this is not as accurate as methods using foot-mounted sensors with ZUPTs every few seconds. Elwell (1999) seems to be the first published researcher to note that each stance phase provides an opportunity to apply a ZUPT; but an earlier unpublished project involving Larry Sher at DARPA also appears to have used similar techniques in 1996 (Sher, 2003)⁶. This is also mentioned by Foxlin (2005).

There are several ways of detecting the best instants to apply a ZUPT. One option is to use knowledge of the human walking pattern to detect the stance phase and then apply the ZUPT. Typically, such methods model walking as a repeating sequence of heel strike, stance, push off and swing (Park and Suh, 2010). The ZUPT is applied during the stance phase. We expect these methods to fail for other modes of movement such as running, crawling or walking backwards. A second option is more generic and tries to determine when the sensor is stationary by using only data from the inertial sensors. The assumption is that, when the sensor is stationary, the measured acceleration is constant and equal to gravity, and the rates-of-turn measured by the gyroscopes are zero. Such methods may incorrectly detect zero-velocity if the sensor moves at constant velocity, and may fail to detect a stance phase if the sensors are very noisy. The occasional failure in ZV detection will increase the accumulated error in the position estimate but will not prevent the inertial tracking system from functioning. Figure 4.2 shows some

⁶<https://dist-systems.bbn.com/projects/PINS/> (Accessed 2012.09.18).

sample sensor data recorded over a few steps.

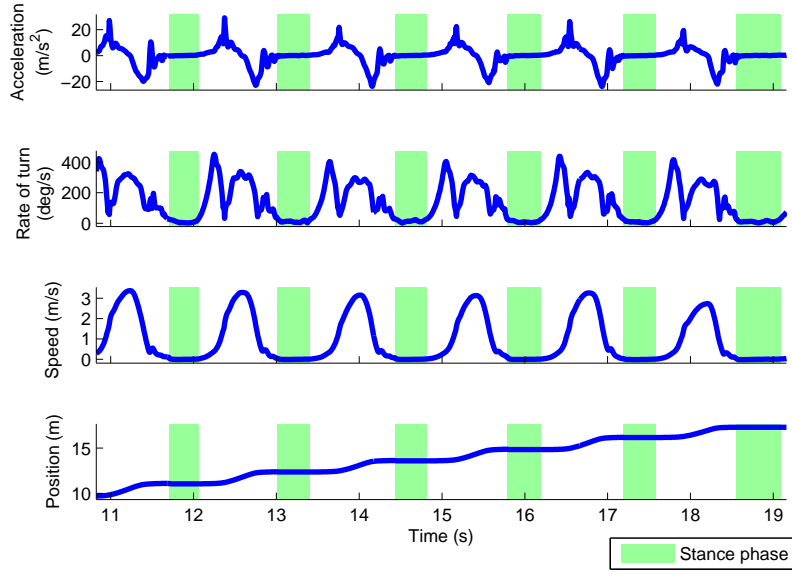


Figure 4.2: PDR algorithm: each step has a stance phase (shaded) and a swing phase. Velocity is reset to zero during the stance phase, acceleration is double integrated during the swing phase.

Researchers have already compared different ZV or stance phase detection methods, often examining the error on the total distance estimate or the error in the final position estimate (Skog et al., 2010b), or looking more closely at the number of steps detected (Callmer et al., 2010). For typical walking, the consensus seems to be that using the gyroscope rates of turn is the most reliable way of detecting ZV. However, Callmer et al. (2010) suggest that including the accelerations in the detection provides better performance when the pedestrian is running. In recent work, Bebek et al. (2010) report using high-resolution pressure sensors under the soles of a boot to detect the phase when the foot is stationary. They achieve slightly better tracking accuracy than when using gyroscopes because they can apply the ZUPTs more accurately. But, generally speaking, based on the results given by the work above and our own experience, different stance phase detection methods tend to have roughly equivalent performance when they are tuned correctly.

4.3.4 Position estimation

Our initial implementation is straightforward. It uses the orientation estimates from the IMU and simple velocity resets. We then improve our results by using a Kalman filter to correct position estimates as well as velocity estimates. Finally, we extend the Kalman filter to compute the orientation directly from the gyroscope and accelerometer measurements, thus giving us more control over our tracking system and even better results.

Naïve implementation

The simplest implementation proceeds in five steps. 1) Transform the accelerations from the sensor frame into the navigation frame using the orientations estimated by the AHRS. 2) Subtract gravity from the vertical axis. 3) Integrate the accelerations to obtain the velocity. 4) Reset the velocity to zero if the sensor is detected to be stationary. 5) Integrate the velocity to obtain the position. We have illustrated this in the flowchart in figure 4.3.

Kalman filter implementation

We improve on the previous method by noting that velocity and position are correlated (Foxlin, 2005). If the estimated velocity is incorrect, it will affect the estimated position in a predictable way. In particular, when we detect that the sensor has stopped moving during a zero-velocity phase, but the estimated velocity is not zero, we know that the position estimate is likely to be incorrect. This means that whenever the sensor is detected to be stationary, we should not only reset the estimated velocity to zero but also adjust the estimated position by a small amount.

We separate the basic inertial navigation system (INS) from the ZUPTs. The INS transforms the accelerations into the navigation frame, subtracts gravity from

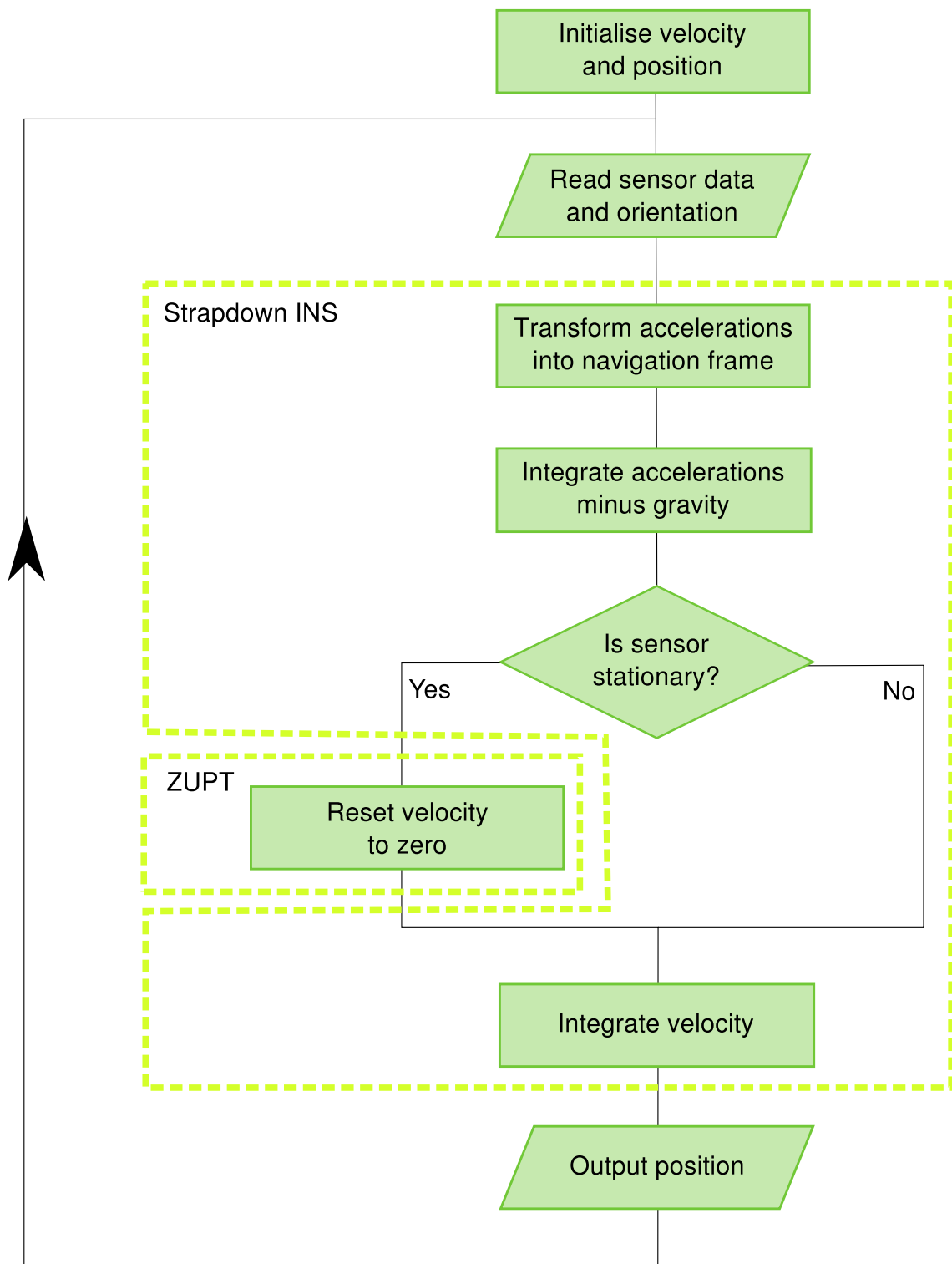


Figure 4.3: Naïve zero-velocity update method.

the vertical axis and integrates twice to obtain the velocity and position estimates. Due to the noisy accelerometer measurements, these basic INS estimates can accumulate several metres of error over a few seconds. But, as in the previous method, ZUPTs can reduce this error to acceptable levels. The most common method used in the literature to implement this correction is the Kalman filter. A Kalman filter estimates the state of a system based on noisy measurements and a system model. In our tracking problem, the *system* is the INS (the foot, the sensor, and the simple integration algorithm); the *state* is the error in velocity and position estimates; the *measurements* are the ZUPTs. In addition to the values of the velocity and position errors, the Kalman filter also estimates their error covariances and cross-covariances. It is the cross-covariances that enable the filter to correct the position (and not only the velocity) during a ZUPT. At the end of each ZUPT, the estimated errors in velocity and position are subtracted from the INS estimates to produce corrected estimates. This is illustrated in figure 4.4.

This method is an error-state, or complementary, Kalman filter, and is a common tool in multi-modal navigation systems. This Kalman filter estimates the deviation from the true state rather than the state itself. The principles are the same as those used in a standard Kalman filter but the implementation looks slightly different. For our application, the implementation is simpler because, as far as the filter is concerned, there is no need to remember the state estimate which is always zero at the beginning of each iteration.

Estimating the orientation of the IMU

As mentioned earlier, commercial IMUs do an excellent job of estimating their orientation, but their AHRS algorithms are complex and usually inaccessible to end-users due to intellectual property issues. One problem we faced is that the AHRS embedded in our XSens MTx IMU performs some online calibration based on the type of movement of the sensor. We noticed that the quality of orientation estimates

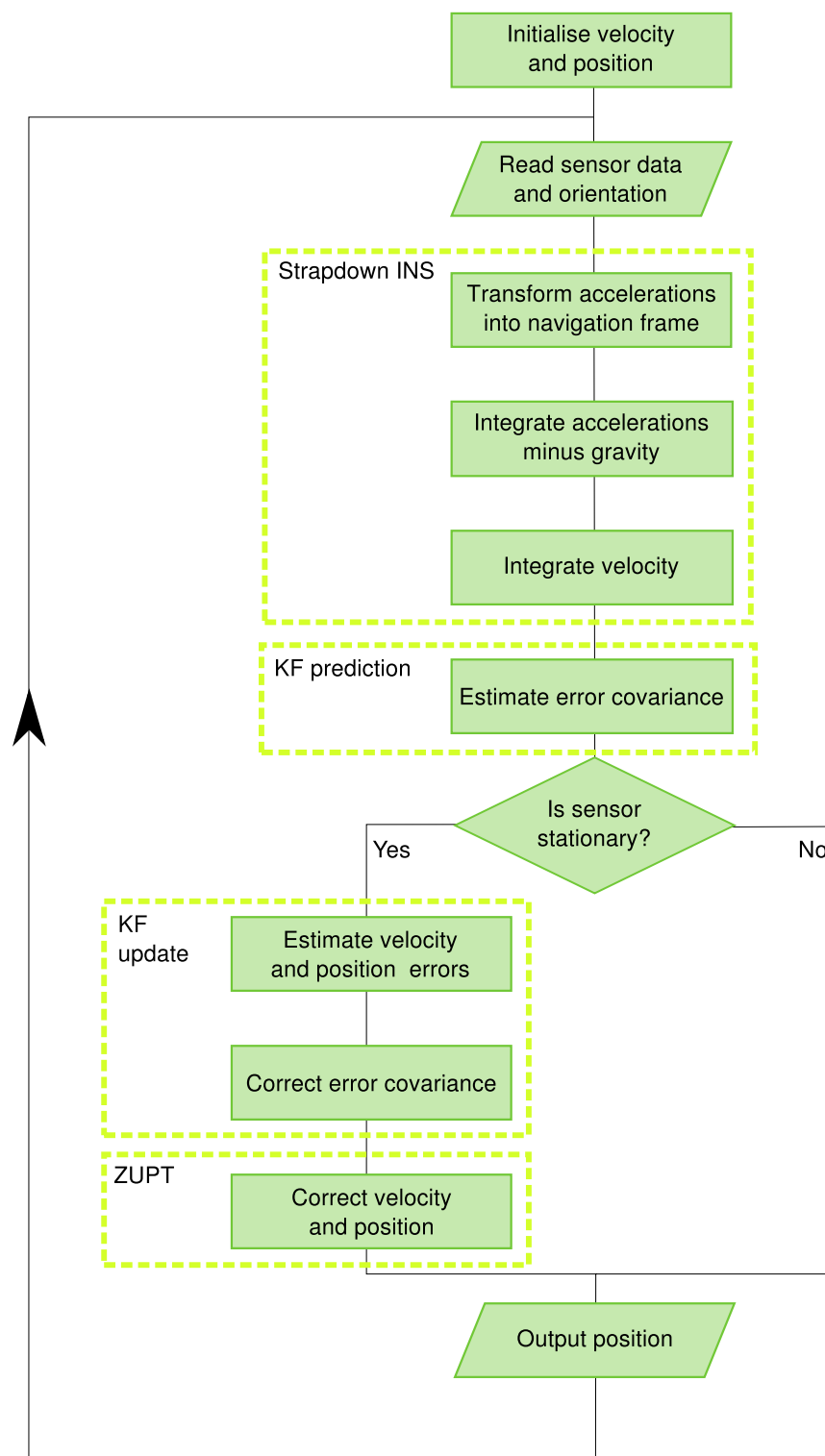


Figure 4.4: Kalman filter zero-velocity update method.

(and therefore the accuracy of tracking) often improves after a few minutes. This suggests that some internal parameters of the AHRS take some time to reach their optimal values. By estimating the orientation of the IMU ourselves, we can optimise the parameters for pedestrian motion, we are no longer dependent on a proprietary algorithm, and we can use other IMUs which may not compute orientation but are less expensive or better quality.

We estimate the orientation by integrating the rates of turn measured by the gyroscopes. The estimated orientations inevitably suffer from drift, but the algorithm corrects them during the ZUPTs. In the same way that the position estimates are correlated with the velocity, so is the orientation. It can therefore be corrected by our Kalman filter even though it is not measured directly. Intuitively, if the orientation is incorrect, the gravity component will not be entirely removed from the measured acceleration. The remaining gravity component will be integrated, and will cause an error in velocity that is correlated with the error in orientation. Thus, there is a strong correlation between the tilt errors (roll and pitch) and the velocity errors because gravity acts on the vertical axis. There is less correlation between yaw (or heading) error and velocity, and therefore less correction of the yaw. In order to minimise yaw drift, we compensate for as much of the constant gyroscope bias as possible by calibrating each sensor prior to use. The estimated biases are the mean of gyroscope readings recorded during several minutes while the IMU is stationary.

The main difference between this method and the previous Kalman filter method is that the orientation must be estimated using the gyroscope readings, and that the orientation error appears in the state vector along with the velocity and position errors. Figure 4.5 describes this algorithm.

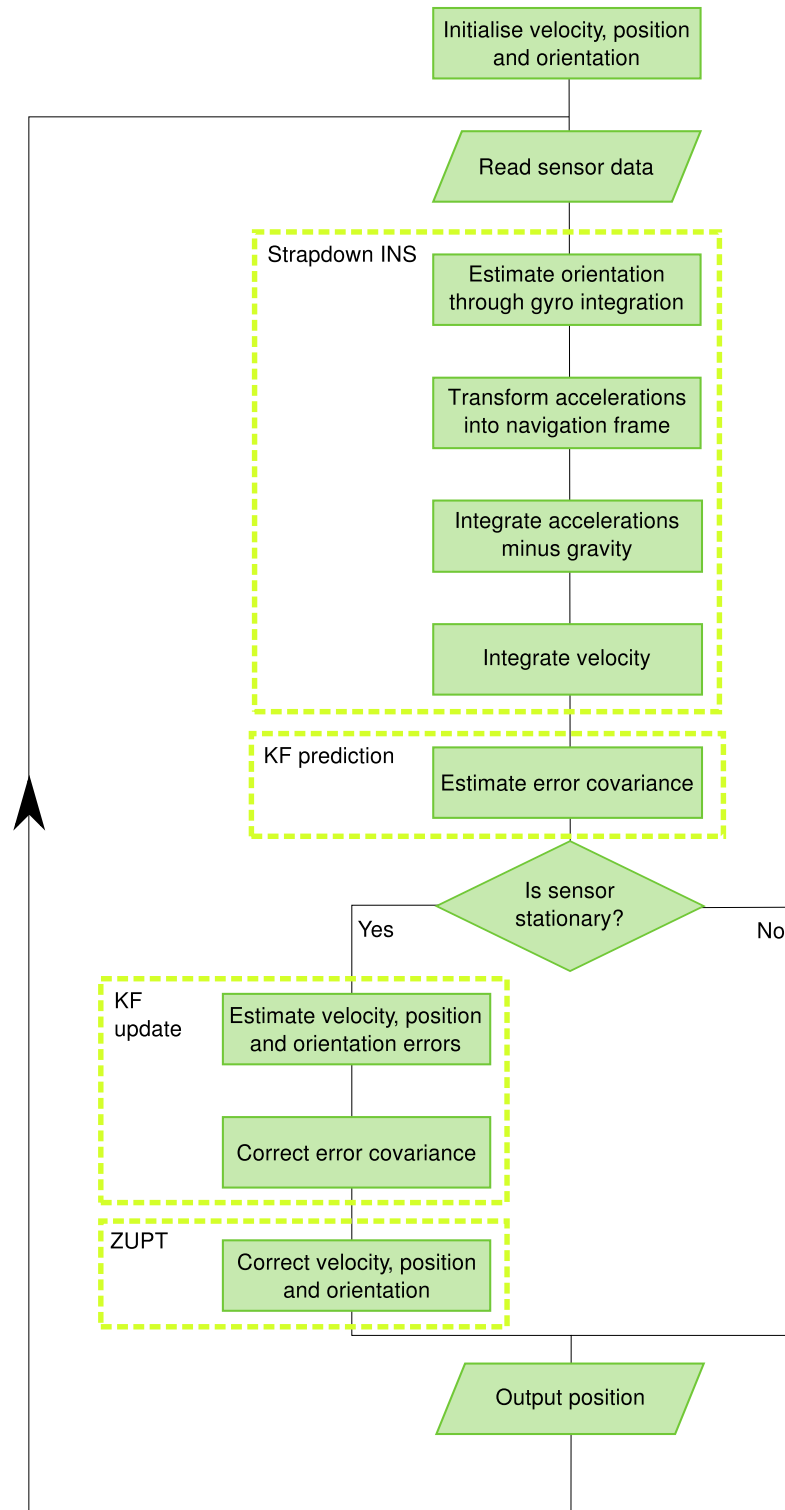


Figure 4.5: Inertial navigation system with orientation estimation corrected by Kalman filter ZUPTs.

4.4 Practical implementation

In the following paragraphs, we give a step-by-step guide to implementing a pedestrian inertial tracking system which uses an error-state Kalman filter for zero-velocity updates and orientation estimation. Although this is a more complex implementation than the straightforward naïve implementation we described first, it takes less than 60 lines of Matlab code and produces much better results. The Matlab implementation is given in appendix A.

4.4.1 Initialisation

Before the main data processing loop, we define parameters, and declare and initialise variables. Table 4.1 (further down) sums up our parameter values.

- Define the ZV measurement matrix $\mathbf{H} = \begin{pmatrix} 0_{3 \times 3} & 0_{3 \times 3} & \mathbf{I}_{3 \times 3} \end{pmatrix}$.
- Define the ZV measurement noise covariance \mathbf{R} as the diagonal matrix with values $\begin{pmatrix} \sigma_{v_x}^2 & \sigma_{v_y}^2 & \sigma_{v_z}^2 \end{pmatrix}$.
- Initialise the position $\mathbf{p} = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}^T$. This value is arbitrary.
- Initialise the velocity $\mathbf{v} = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}^T$. The sensor is assumed to be stationary; if it is moving, the first step estimate will be incorrect.
- Initialise the error covariance matrix $\mathbf{P} = \mathbf{0}_{9 \times 9}$.

- Initialise the orientation matrix \mathbf{C} based on the accelerations. Here, we assume the sensor is stationary. The yaw value is arbitrary.

$$\begin{aligned}
 roll &= \arctan(a_y^{sensor}/a_z^{sensor}) \\
 pitch &= -\arcsin(a_x^{sensor}/g) \\
 yaw &= 0 \\
 \mathbf{C} &= \begin{pmatrix} \cos(pitch) & \sin(roll)\sin(pitch) & \cos(roll)\sin(pitch) \\ 0 & \cos(roll) & -\sin(roll) \\ -\sin(pitch) & \sin(roll)\cos(pitch) & \cos(roll)\cos(pitch) \end{pmatrix}
 \end{aligned}$$

4.4.2 Zero-velocity detection

Apply the ZV test to each data sample within the main loop. We achieved good results using the detection method described by Jiménez et al. (2010). They apply thresholds to the acceleration magnitude, gyroscope magnitude and local acceleration variance. However, a simple threshold on the magnitude of the gyroscope rate-of-turn measurements also works well; a ZUPT is applied when $\|\omega_k\| < \alpha_\omega$, where ω_k is the gyroscope measurement vector at timestamp k and α_ω is the chosen threshold for detecting zero velocity. This is confirmed by Skog et al. (2010a) who show that their gyroscope-based detector works as well as the one using both accelerometers and gyroscopes for typical walking.

4.4.3 Main loop

The following operations are performed for every measurement sample. k is the sample index and is occasionally omitted to simplify notations. Note that this is an iterative algorithm where the estimated orientation of the sensor is used to process the current measurement and re-evaluate the orientation.

1. Compute the time step Δt from the previous measurement. Δt is equal to the sampling interval which is typically constant.
2. Subtract the gyroscope bias from the measurements if required.
3. Compute the skew-symmetric angular rate matrix $\mathbf{\Omega}$ from the gyroscope readings: $\mathbf{\Omega}_k = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}$. This matrix represents an infinitesimal rotation as measured by the gyroscope.
4. Update the orientation matrix: $\mathbf{C}_k = \mathbf{C}_{k-1} (2\mathbf{I}_{3 \times 3} + \mathbf{\Omega}_k \Delta t) (2\mathbf{I}_{3 \times 3} - \mathbf{\Omega}_k \Delta t)^{-1}$ (Qi and Moore, 2002). Postmultiply \mathbf{C} by the update factor because the gyroscope measurements are taken in the sensor frame.
5. Transform the measured accelerations from the sensor frame into the navigation frame: $\mathbf{a}_k^{nav} = (\mathbf{C}_k + \mathbf{C}_{k-1}) \mathbf{a}_k^{sensor} / 2$. Use the average of the previous and the current orientation estimate because the movement has taken place over the time interval between the measurements (as a rough approximation).
6. Integrate the acceleration in the navigation frame minus gravity to obtain the velocity estimate: $\mathbf{v}_k = \mathbf{v}_{k-1} + \left(\mathbf{a}_k^{nav} + \mathbf{a}_{k-1}^{nav} - 2 \begin{pmatrix} 0 & 0 & g \end{pmatrix}^T \right) \Delta t / 2$. Use the trapeze method of integration.
7. Integrate the velocity to obtain the position estimate:

$$\mathbf{p}_k = \mathbf{p}_{k-1} + (\mathbf{v}_k + \mathbf{v}_{k-1}) \Delta t / 2.$$
8. Construct the skew-symmetric cross-product operator matrix \mathbf{S} from the navigation frame accelerations: $\mathbf{S}_k = \begin{pmatrix} 0 & -a_z^{nav} & a_y^{nav} \\ a_z^{nav} & 0 & -a_x^{nav} \\ -a_y^{nav} & a_x^{nav} & 0 \end{pmatrix}$. This matrix relates the variation in velocity errors to the variation in orientation errors.

9. Construct the state transition matrix:
$$\mathbf{F}_k = \begin{pmatrix} \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{I}_{3 \times 3} & \mathbf{I}_{3 \times 3} \Delta t \\ -\mathbf{S}_k \Delta t & \mathbf{0}_{3 \times 3} & \mathbf{I}_{3 \times 3} \end{pmatrix}.$$

10. Construct the process noise covariance matrix \mathbf{Q}_k as the diagonal matrix with values $\left(\begin{pmatrix} \sigma_{\omega_x} & \sigma_{\omega_y} & \sigma_{\omega_z} & 0 & 0 & 0 & \sigma_{a_x} & \sigma_{a_y} & \sigma_{a_z} \end{pmatrix} \Delta t \right)^2$.
11. Propagate the error covariance matrix $\mathbf{P}_k = \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T + \mathbf{Q}_k$.
12. Detect a stationary phase when $\|\omega_k\| < \alpha_\omega$.

The following operations are only performed for samples occurring in a stationary phase.

13. Compute the Kalman gain $\mathbf{K}_k = \mathbf{P}_k \mathbf{H}^T (\mathbf{H} \mathbf{P}_k \mathbf{H}^T + \mathbf{R})^{-1}$.
14. Compute state errors from the Kalman gain and estimated velocity: $\boldsymbol{\varepsilon}_k = \begin{pmatrix} \boldsymbol{\varepsilon}_C & \boldsymbol{\varepsilon}_p & \boldsymbol{\varepsilon}_v \end{pmatrix}^T = \mathbf{K}_k \mathbf{v}_k$. Here, the complete error vector $\boldsymbol{\varepsilon}$ is composed of the three elements of the attitude error (error on roll, pitch and yaw angles), the position error, and the velocity error, in that order.
15. Correct the error covariance: $\mathbf{P}_k = (\mathbf{I}_{9 \times 9} - \mathbf{K}_k \mathbf{H}) \mathbf{P}_k$ (alternatively, use the more robust Joseph form (Simon, 2006, eq. 5.19)).
16. Construct the skew-symmetric correction matrix for small angles: $\boldsymbol{\Omega}_{\varepsilon,k} = \begin{pmatrix} 0 & \boldsymbol{\varepsilon}_C[3] & -\boldsymbol{\varepsilon}_C[2] \\ -\boldsymbol{\varepsilon}_C[3] & 0 & \boldsymbol{\varepsilon}_C[1] \\ \boldsymbol{\varepsilon}_C[2] & -\boldsymbol{\varepsilon}_C[1] & 0 \end{pmatrix}$. The indices are one-based, so $\boldsymbol{\varepsilon}_C[1]$ is the first element of the attitude error (roll).
17. Correct the attitude estimate: $\mathbf{C}_k = (2\mathbf{I}_{3 \times 3} + \boldsymbol{\Omega}_{\varepsilon,k}) (2\mathbf{I}_{3 \times 3} - \boldsymbol{\Omega}_{\varepsilon,k})^{-1} \mathbf{C}_k$ (Qi and Moore, 2002). Premultiply \mathbf{C} by the correction factor because the correction is computed in the navigation frame.
18. Correct the position estimate: $\mathbf{p}_k = \mathbf{p}_k - \boldsymbol{\varepsilon}_p$.
19. Correct the velocity estimate: $\mathbf{v}_k = \mathbf{v}_k - \boldsymbol{\varepsilon}_v$.

4.5 Evaluation

We do not aim to give a precise comparison of different PDR techniques, or to determine optimal parameters. Rather, we want to provide a practical guide to what performance can be expected from the standard implementation we have described.

Detailed ground truth, with both position and timestamps, is difficult to obtain, but even when it is available, aligning it with the estimated path, and computing position errors remains a challenge. All position estimates are relative to the initial position and heading, and a small error early in the path can have a significant effect later on, even if no further errors occur.

In the literature, researchers have used several ways to evaluate the performance of PDR systems.

- Inspection of estimated paths in the coordinate space and comparison to simple geometric ground truth.
- Computing the distance between starting point and final position estimate for a closed loop walk; smaller distances indicate less drift; this does not account for scaling errors.
- Computing the estimated total distance traveled; this will give different results depending on whether distance is accumulated per measurement or per step, and does not account for heading errors.
- Computing the distance and heading error for each step (or segment) by realigning each previous step (or segment) to the ground truth, as suggested by Angermann et al. (2010).

The quality of ZV detection can be evaluated by comparing ZV detection using the IMU to ZV detection using force sensitive resistors on the sole of the footwear (Bebek et al., 2010), or video recordings of the foot movement (Skog et al., 2010a). Our goal in this chapter is not to fine tune the tracking algorithm but simply to give a clear and concise implementation. Therefore, rather than providing a quantitative evaluation of our work, we illustrate its performance by plotting and manually aligning the position estimates on a floorplan. In our figures, the ground truth is visible in the outlines of the corridors and the location of the stairs which are clearly visible both on the floorplan and in the plotted path.

4.5.1 Recordings at Lancaster University’s Infolab21

We tested our pedestrian tracking system with six different people. They walked approximately 240 metres in our office building, including several flights of stairs, taking approximately four minutes. We used an XSens MTx sensor (model MTx-28A53G25 or MTx-49A53G25) to record inertial measurements at 120 samples per second. These sensors are the standard model with accelerometers with a full scale of $\pm 50 \text{ m/s}^2$ and gyroscopes with a full scale of ± 1200 degrees per second. We have since realised that the accelerometer range is insufficient for optimal tracking; the $\pm 180 \text{ m/s}^2$ model would have been more suitable. The sensor was attached to the instep of the foot with a velcro strap (figure 4.6). In our experiments, the exact position of the sensor made little difference to the results. Other researchers have attached it to the heel or embedded it into the sole of a boot. The pattern of accelerations and rotations is different for each position, and this should affect the accuracy of the zero-velocity detection and of the tracking, but we hypothesise that this is a small source of error compared to the gyroscope yaw bias for instance.

Figure 4.7 shows the horizontal and altitude plots from our three implementations: the naïve implementation and the Kalman filter implementation both using



Figure 4.6: IMU attached to the instep of the foot with a velcro strap. The wire goes up inside the trouser leg so it does not interfere with walking.

the orientations estimated by the built-in AHRS, and the Kalman filter implementation estimating the orientation itself. The fourth pair of plots shows the improvement in horizontal position estimates when we manually correct the gyroscope bias. The plots chosen for this chapter illustrate the improvement achieved for the altitude estimates by using the Kalman filter for the ZUPTs, and the additional improvement in horizontal position estimates when we compute the orientations ourselves and compensate for gyroscope bias.

There was a substantial difference between different subjects and sensors. Gyroscope bias varies between sensors and also between switch-ons of the same sensor. Also, due to the dynamic range of our accelerometers being slightly too small (an unfortunate choice of hardware), the walking patterns of some subjects introduced more data clipping than others. For some subjects, we were unable to achieve such accurate altitude estimates (drift of up to four metres over the whole walk) as shown in figure 4.7. On the other hand, some datasets did not require any gyroscope bias correction and produced very accurate horizontal plots even with the simpler implementations. Nevertheless, the Kalman filter implementation with orientation estimation and manual gyroscope bias correction always produced the best results. The stairs are clearly visible at both ends of the building, as well as two detours to avoid seating areas, and even a small kink in the middle of a corridor where the subject paused to open a door. Data recorded at 120 samples per

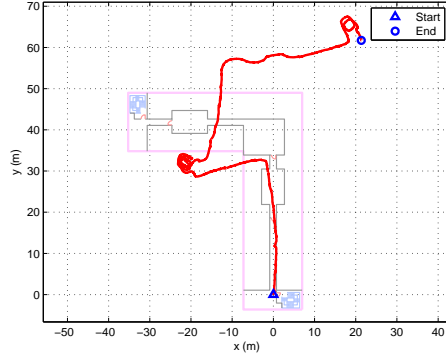
second gave the best results; lower sample rates displayed degraded performance, but higher rates did not bring any noticeable improvement.

4.5.2 DLR figure-of-eight recording

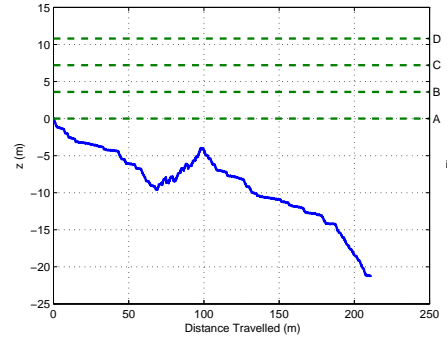
Figure 4.8 is the output from our Kalman filter implementation for one of the datasets recorded by the team at the German Aerospace Centre (DLR) (Angermann et al., 2010), and shows that our proposed algorithm works well for other data than ours. Michael Angermann and colleagues recorded inertial measurements and ground truth for sixteen different walking experiments in a lab equipped with a high precision motion capture system. They use an identical sensor to ours, the Xsens MTx (model 28A53G25), sampled at 100 samples per second. They do not provide the orientation estimates computed by the IMU so we only show the horizontal and altitude estimates for the Kalman filter implementation with orientation estimation. After manual alignment, the estimated path remains within 30cm of the ground truth, and the altitude only drifts by 50cm over a total distance of 50m.

4.5.3 Running

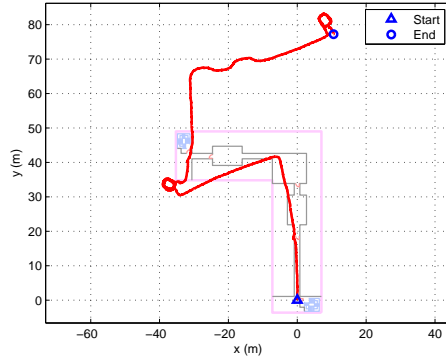
Figure 4.9 shows the path estimated by our PDR implementation when the pedestrian ran along a corridor. The implementations using the orientation estimated by the AHRS give very poor results and are not shown here. When estimating the orientation in our algorithm, the horizontal estimate is acceptable but not the altitude estimate. We believe the latter would improve considerably if the accelerometers had a wider measurement range and could accurately record the high accelerations on impact.



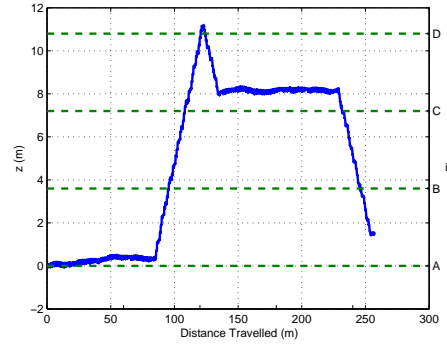
(a) Naïve ZUPT: horizontal position.



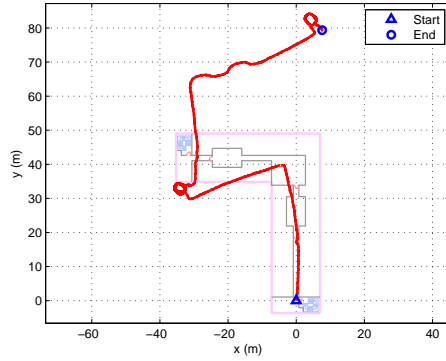
(b) Naïve ZUPT: altitude.



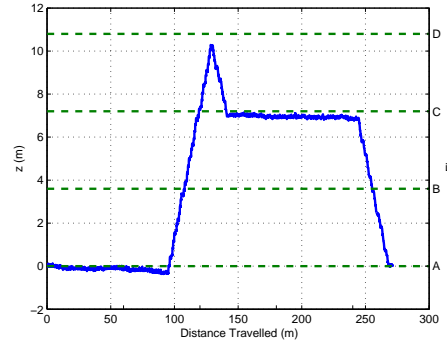
(c) KF ZUPT: horizontal position.



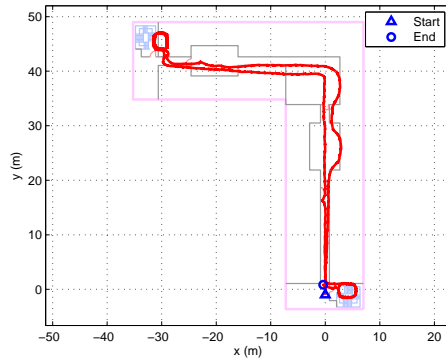
(d) KF ZUPT: altitude.



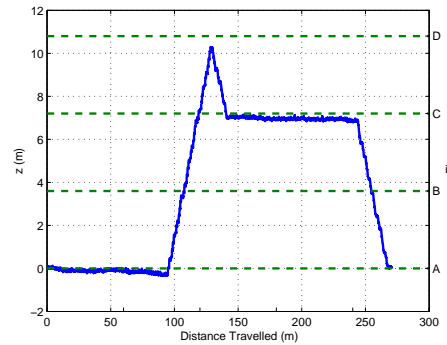
(e) KF with orientation estimation: horizontal position.



(f) KF with orientation estimation: altitude.



(g) KF with orientation estimation and gyroscope bias correction: horizontal position.



(h) KF with orientation estimation and gyroscope bias correction: altitude.

Figure 4.7: PDR estimated path of a four minute walk through the Infolab, including stairs at both ends of the building. The stairs, outline of the corridor, and height of the different floors make the true path apparent, particularly on the last figure.

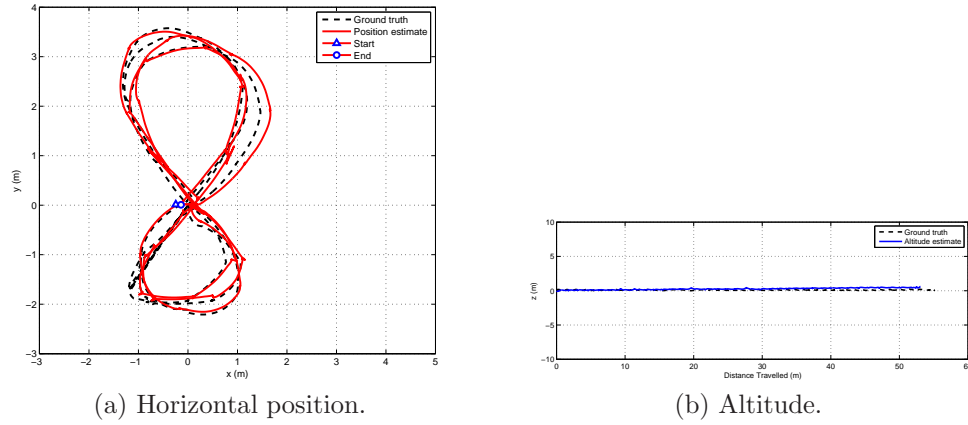


Figure 4.8: PDR estimated path of the DLR figure-of-eight data set — KF implementation with orientation estimation.

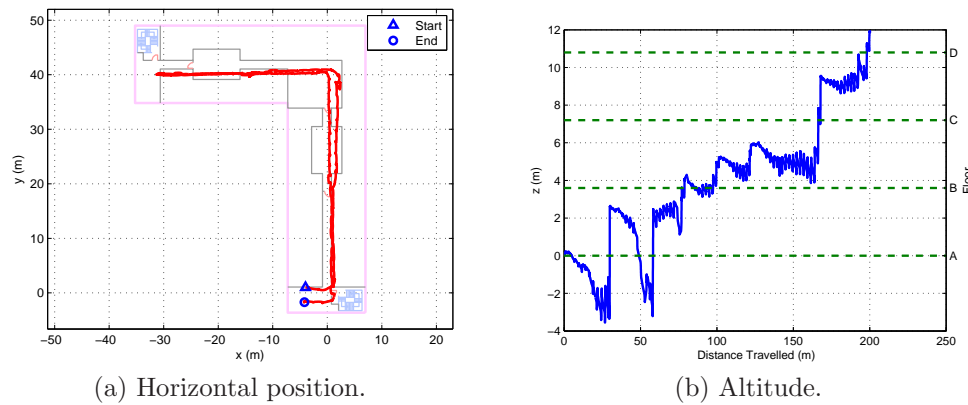


Figure 4.9: Running along a corridor in the Infolab. KF implementation with orientation estimation and gyroscope bias correction. The true path is approximately 45.6×34.5 metres in a horizontal plane, returning to the starting point.

4.6 Challenges and suggestions

Two years ago, we started investigating pedestrian inertial tracking as a component of a larger multimodal indoor localisation system. Since then, we have moved from the naïve implementation described first in this chapter, to the Kalman filter implementation, and more recently to estimating the orientation in the algorithm itself. Each new implementation brought a better understanding of the inner workings of pedestrian inertial tracking. In the following paragraphs, we share some of the lessons we have learnt.

4.6.1 Major causes of tracking error

The gyroscope bias was the cause of most of the horizontal errors observed during our work. It is only possible to compensate for these biases in the final implementation of the tracking filter, in which we estimate the orientations from the raw inertial sensor data, hence the better results in the final plots of figure 4.7. We now recalibrate our gyroscopes before each experiment by recording several minutes of data while the sensor is not moving. The mean of the gyroscope readings for each axis gives us the bias which needs to be subtracted from all measurements before running the PDR algorithm. This method works well even if the sensor is attached to the foot of a pedestrian during calibration. This is a run-specific calibration which should be performed again each time the sensors are powercycled, as the bias changes from switch-on to switch-on. We have been unsuccessful in our attempts to perform online calibration, that is the estimation of the sensor biases while the system is running. Although others have included gyroscope and accelerometer bias estimation in their pedestrian inertial navigation systems (Bebek et al., 2010; Foxlin, 2005), we have not managed to improve performance with such methods. We believe it is for the following reasons. Much of the accelerometer bias is already compensated for during the zero velocity updates. As we mentioned

earlier, there is little correlation between yaw (heading) and velocity. For the same reason, there is little correlation between the gyroscope bias corresponding to the yaw, and the velocity (see 4.3.4 for more details). Thus, the most significant bias could only be estimated with additional sensor inputs, not solely with zero velocity updates. For instance, this would be feasible if we had position estimates from a GPS receiver. In theory, we could also use readings from a magnetometer to correct the heading and estimate the gyroscope yaw bias, but our attempts at this were unsuccessful, primarily due to the magnetic interference caused by nearby large metallic objects.

The other type of error we encounter frequently is altitude error. Faulkner et al. (2010) give an analysis of the altitude errors in inertial pedestrian tracking systems. The authors found that they are due to the accelerations of the foot exceeding the dynamic range of the accelerometers. They noted that accelerations can reach $\pm 10g$ when walking, and $\pm 13g$ when descending stairs or running. This confirms our own observations, and explains why the quality of our altitude estimates varies so much between users with our $5g$ accelerometers. For the best results, developers should use accelerometers with at least a $\pm 10g$ range and gyroscopes with a $\pm 900^\circ/s$ range. Currently, these are close to the highest specifications available for MEMS inertial sensors.

4.6.2 Parameter tuning

It is reasonable to tune the system parameters to some extent based on sensor characteristics and prior knowledge of how the system will be used. However, it should perform as well as possible for a variety of different types of movement and without any user-specific training. The choice of one parameter value will affect some of the others. In some cases, a poor choice of one parameter or even an error in the implementation can mask another incorrect parameter. Table 4.1 gives a list of parameter values used in our final implementation. Others should

get acceptable results with these values but may be able to improve with further tuning. Nilsson et al. (2010) give a more formal analysis of the effects of different parameters on performance. Note that due to an error, the values they give for the noise parameters should be divided by 250 (the sampling frequency).

Step 10 of the algorithm, which estimates the process noise \mathbf{Q} , simply multiplies the gyroscope and accelerometer noise (σ_ω and σ_a) by the timestep in order to determine the orientation and velocity noise, and sets position noise to zero. Note that these noise values take into account all sources of error in the INS, not only short term sensor noise, but also bias variations, scaling errors and integration errors. They are therefore different from noise values reported in sensor datasheets.

The zero-velocity measurement noise σ_v , used to construct \mathbf{R} , represents the uncertainty in velocity during a ZUPT and should therefore be one or two orders of magnitude smaller than an average walking speed of 2 m/s. A more precise ZV detector would require a smaller value, and a less precise one a larger value. Generally, if the ratio of \mathbf{Q}/\mathbf{R} remains constant, the system will perform the same. We recommend setting \mathbf{R} , then adjusting \mathbf{Q} .

We found an approximate value for the gyroscope ZV detection threshold α_ω by plotting the norm of gyroscope rates-of-turn over time for a sample pedestrian recording. Periodic stance phases are easy to recognise in such a plot. We set the threshold value slightly higher than the value of the norm during these phases and then adjusted it to get the best tracking results on the sample recording.

4.6.3 Output format

If the system is going to be used alone, it can simply output the estimated Cartesian coordinates of the pedestrian. If the PDR system is just one component of a larger localisation system, it may be more convenient to output incremental values such

Table 4.1: PDR system parameter values.

Name	Notation	Value
Time step	Δt	1/120s
Accelerometer noise	σ_a	0.01 m/s^2
Gyroscope noise	σ_ω	0.01 rad/s
Gravity	g	9.8 m/s^2
ZV measurement noise	σ_v	0.01 m/s
Gyroscope ZV detection threshold	α_ω	0.6 rad/s

as step length, variation in direction of travel, and variation in altitude. This allows easier integration with other localisation systems such as covered by Groves (2008). Some applications may not require or be able to process position updates at the high sampling rate of the inertial sensors. In this case, the PDR system should output estimates only once per step, or at a fixed rate.

4.6.4 Tracking the orientation

We found it useful to track the yaw of the IMU as well as the direction of travel. This allows us to distinguish between the pedestrian walking forward or backward, and is particularly useful in applications where we want to orient a display according to the direction the user is facing (rather than the direction they are walking in). Assuming the x-axis of the IMU is aligned with the *forward* direction of the pedestrian, we can easily compute the yaw from the orientation matrix as $yaw = \arctan(C_{2,1}/C_{1,1})$. Another option which we explored briefly is to attach an additional inertial measurement unit to the torso or head of the pedestrian. We included its position and orientation estimates in the Kalman filter in the same way as the foot-mounted sensor but without the ZUPTs. By regularly feeding the filter artificially generated pseudo-measurements which indicate that these new sensors are a fixed distance above the foot-mounted sensor (this is an approximation), the filter can estimate the orientation of the torso or the head.

4.6.5 Common mistakes

During our development of PDR systems, we often committed several trivial errors which we list here to help others. When updating or correcting the orientation matrix, developers must take care to post-multiply or pre-multiply as appropriate. Notations and definitions vary between publications, and we have noted small errors which completely change the meaning of an equation.

We also committed a more fundamental error by subtracting the acceleration due to gravity from the measured acceleration immediately after transforming it into the navigation frame. The acceleration due to gravity should indeed be removed during the integration phase. However, it should be left untouched when constructing the skew-symmetric matrix \mathbf{S} , in order to preserve the correlation between orientation and velocity.

4.7 Conclusion

In this chapter, we have shown how to implement a simple inertial pedestrian dead reckoning system with comparable performance to other current implementations. We have highlighted the requirements in terms of sensor specifications and given some advice on selecting a good set of parameters for the system. During our work on this algorithm, we were constantly reminded of the inherent limitation of PDR, namely positional drift. This drift introduces error in position estimates, either suddenly, due to a hardware glitch or an out-of-range measurement, or gradually, due to slowly accumulating measurement errors.

This shortcoming of PDR makes it unreliable as a standalone tracking system, but still very powerful as a component of a hybrid system. In our search for robust localisation in unknown and uninstrumented environments, we believe we need to

be more creative in our use PDR. Many PDR-based systems sacrifice the primary benefits of PDR by relying on some form of infrastructure (e.g., wifi, RFID tags) or prior knowledge (e.g., maps and floorplans). We have found only a few systems which are careful to preserve the advantages of PDR. FootSLAM (Robertson et al., 2009) constructs a map of walkable areas in a building using only PDR traces. More recently, it has been improved by allowing approximate locations to be tagged, either automatically from some external detection mechanism, or manually by the user (Robertson et al., 2010). Another approach to improving estimates is to combine traces from multiple pedestrians. We have contributed to some work which shows that this works on a very large scale with tens or hundreds of users (Kloch et al., 2011), all carrying PDR-enabled mobile phones which detect their proximity to each other using Bluetooth. Others have shown that combining position estimates from only two pedestrians (Strömbäck et al., 2009) or from several sensors on the same pedestrian (Jin et al., 2011) also improve accuracy. We would like to see more research projects like these which acknowledge the strengths of PDR and the need for systems which can support navigation in challenging environments.

The final avenue of research which could address the issue of drift is improving the hardware itself. The specially-designed NavChip (Wan and Foxlin, 2010) by Intersense⁷ offers some improvements over other generic inertial sensors when it comes to PDR but still suffers from the same fundamental limitations. Sysnav⁸ address the issue of localisation in GPS-denied environments by using an array of magnetometers to estimate displacements. In his thesis, Dorveaux (2011) demonstrates how this magneto-inertial sensing produces similar results to our inertial PDR without requiring the sensors to be foot-mounted. A major shift in technology could offer a solution if it were to drastically reduce the magnitude of inertial measurement noise, especially bias.

Inertial pedestrian dead-reckoning provides a suitable starting point for many,

⁷<http://intersense.com> (Accessed 2012.09.24.)

⁸<http://sysnav.fr> (Accessed 2012.09.24.)

more elaborate, hybrid localisation systems, which could provide a viable navigation solution for emergency responders or others who need support in unknown and uninstrumented environments. In the following chapter, we use PDR to initialise the position estimates of sensor nodes which will then provide the robustness that PDR alone is lacking.

Simultaneous localisation and mapping using pedestrian dead reckoning and ultrasonic sensor nodes¹

In chapter 3, we described how ultrasonic sensor nodes could conceivably be used as way-finding beacons during a firefighting intervention. A trail of these sensors could be used to provide navigation support. However, their limited range and accuracy, and the possibility that some would be moved or even destroyed, would make this solution cumbersome and unreliable. As shown in chapter 4, pedestrian dead reckoning is more compact and tracks with more detail than ultrasound, but it intrinsically suffers from drift and cannot be relied upon for more than a few minutes. In this chapter, we combine these two technologies through a technique known as simultaneous localisation and mapping (SLAM). The ultrasonic sensor nodes act as landmarks to correct the drift in pedestrian dead reckoning (PDR), and PDR is used to initialise and refine the positions of the landmarks while

¹This chapter is a revised version of: Carl Fischer, Kavitha Muthukrishnan, and Mike Hazas. SLAM for Pedestrians and Ultrasonic Landmarks in Emergency Response Scenarios. In Marvin V. Zelkowitz, editor, *Advances in Computers*, volume 81, chapter 3, pages 103–160. Academic Press, Burlington, MA, 2011. The author acknowledges that the data collection and algorithm development were conducted in collaboration with Dr Kavitha Muthukrishnan.

providing detailed tracking even where sensor nodes are missing. Such a system has the potential to provide accurate and robust localisation for pedestrians in unknown environments.

5.1 Introduction

Our motivation for SLAM is to overcome the need for an *a priori* map and pre-installed infrastructure, and to enable mapping and navigation that is both extensible and adaptive in a changing environment. There is a lot of existing research to draw on, and yet we found very little on pedestrian SLAM at the time of writing. This chapter focuses on the creation and evaluation of a pedestrian SLAM system in a controlled lab setting as a first step towards deploying such a system in the real world.

5.1.1 Related work

SLAM has been extensively studied in the field of robotics (Djugash et al., 2006; Dissanayake et al., 2001) and has been specifically applied to a variety of environments such as indoor (Djugash et al., 2005), outdoor (Guivant et al., 2000), aerial (Kim and Sukkarieh, 2007) and undersea (Newman and Leonard, 2003; Olson et al., 2006). We report briefly some of the work developed within the context of robotic SLAM and how there is now an interest in SLAM for pedestrians. For a more comprehensive review of SLAM, we refer the reader to Thrun (2002), Durrant-Whyte and Bailey (2006), and Bailey and Durrant-Whyte (2006).

SLAM in robotics

The basic setting for the SLAM problem is a robot with a known kinematic model starting at an unknown location, and moving through an unknown environment

containing landmarks. The robot is equipped with sensors that can detect and measure the location of these landmarks relative to itself. For instance, these sensors can be a camera to detect visual features (Folkesson and Christensen, 2008; Andreasson et al., 2007), a laser range finder to observe the shape of surrounding obstacles such as walls or trees (Hähnel et al., 2003), an RFID reader to detect tags (Hähnel et al., 2004), or an ultrasound sensor to measure the range and bearing to special beacons (Djugash et al., 2006). The SLAM algorithm places the landmarks on a map, as the robot's sensors observe them, using the robot pose estimate (i.e., position and orientation) to determine their locations in the map. The algorithm simultaneously uses landmark observations to refine both the robot's pose estimate and the landmark location estimates. As the landmarks are repeatedly observed, the confidence in their location estimates increases, and the map converges to an accurate representation.

Some of the notable problems that are widely researched by the SLAM community are: (i) the *complexity* of the SLAM methods, which becomes an issue when the number of landmarks in the map increases, and (ii) *data association* of the observed landmarks with landmarks held in the map. This second point is particularly important for *loop closure*, when a robot returns to a previously mapped region after traversing a long path. These problems are being addressed in detail within robotics research (Durrant-Whyte and Bailey, 2006; Bailey and Durrant-Whyte, 2006).

Approaches to SLAM

SLAM methods fall mainly in three categories: (i) EKF-SLAM, which employs an extended Kalman filter (EKF) to represent the joint state space of robot pose and all landmarks that have been identified (Smith and Cheeseman, 1986); (ii) FastSLAM, which uses a Rao-Blackwellized particle filter in which each particle

effectively represents a pose and set of independent compact EKF's for each landmark (Thrun et al., 1998); and (iii) GraphSLAM, which models the landmarks and the successive positions of the target as nodes of a graph, and measurements as edges of this graph (Thrun and Montemerlo, 2006).

The EKF approach has two serious drawbacks that prevent it from being applicable to certain large-scale real environments. Firstly, the complexity is quadratic in the number of landmarks, thus limiting the number of landmarks that can be handled by this approach. Secondly, it relies heavily upon the assumption that the mapping between observations and landmarks is known. Associating a small number of observations with incorrect landmarks in the EKF can cause the filter to diverge. This shortcoming has been recognised and investigated by the community (Guivant and Nebot, 2001; Leonard and Feder, 2000). The computational effort has been reduced by using submapping methods — splitting the global map into a number of submaps (Leonard and Feder, 2000) — and by using sparse information matrices instead of covariance matrices (Thrun et al., 2004).

FastSLAM was introduced by Thrun et al. (1998) and Montemerlo et al. (2002) as a more efficient SLAM algorithm. FastSLAM decomposes the SLAM problem into a robot localisation problem and several landmark estimation problems that are conditioned on the robot's pose estimate. FastSLAM uses a particle filter for estimation. Each particle effectively represents a pose and a set of independent low-dimensional EKF's, one for each landmark. The conditioning on a pose allows the landmarks to be estimated independently, thus lowering the complexity. Further research by Montemerlo (2003) showed that FastSLAM is able to deal with ambiguous data association more reliably than EKF methods. This approach does have some drawbacks which are explored by Bailey et al. (2006). In particular, the uncertainty estimates inevitably become too optimistic and the filter is unable to explore the complete state-space. A further publication by Brooks and Bailey (2009) present a hybrid approach combining the benefits of EKF-SLAM and FastSLAM.

Graph-based methods represent the positions of the landmarks and the successive positions of the target (e.g., robot or pedestrian) as nodes in a graph. Odometry or dead-reckoning measurements become edges between target positions, and landmark observations (e.g., range measurements) are edges between target positions and landmark positions. The positions of the nodes can be estimated using optimisation methods such as gradient descent or MDSMAP (Shang and Ruml, 2004). Kleiner and Sun (2007) use a graph representation to build a map of landmarks (the nodes of the graph) after several pedestrians have walked between them (thus measuring the edges of the graph). Djughash et al. (2006) use a similar representation to initialise landmark positions but then use EKF-SLAM to track their robot. Golfarelli et al. (1998) refer to their own method as “elastic correction” because it models the connections between nodes as springs which constrain their positions. Thrun and Montemerlo (2006) give a formal description of GraphSLAM. They present it as a solution to the *off-line* SLAM problem for large environments. In other words, it can only be used after all the data has been collected, not in real-time. Off-line solutions such as GraphSLAM are able to produce more accurate maps and traces than on-line methods such as EKF-SLAM and FastSLAM, because they have more data to work with. Thrun (2001) use FastSLAM for real-time tracking, but perform a backward correction of past poses whenever the system detects a loop in the path. This backward correction aims to optimise the consistency of the map similar to other graph-based methods.

Typical sensing modalities for SLAM

Dead reckoning traces its roots back to ship navigation. It is a common technique in robotics where it is often implemented using wheel odometry, but it is now becoming popular in the area of pedestrian tracking due to the miniaturisation of inertial sensors. This method tracks the movements of the target (vehicle, robot, or person) and deduces their current position relative to a previous position. Not all

SLAM methods rely on wheel-based odometry or inertial dead-reckoning. Hähnel et al. (2003) match successive scans from a laser scanner in order to estimate motion. Tardif et al. (2008) use a similar method, *visual odometry*, to estimate the trajectory of a single camera using only a stream of images.

In SLAM with range-only sensors (Newman and Leonard, 2003; Olson et al., 2006; Blanco et al., 2008a) or bearing-only sensors (Deans and Hebert, 2000), a single measurement does not contain enough information to estimate the location of a landmark. Particle filters and Kalman filters can handle this without any particular modifications once a landmark has been initialised. However, in the case of the EKF, in order to initialise it, we must observe it from multiple vantage points. This is one reason why some researchers have preferred to use a stereo camera rig which provides depth information as well as bearing (Davison, 1998). With a single camera, we must collect measurements over a period of time, and initialise the landmark using a batch update scheme (Durrant-Whyte and Bailey, 2006; Bailey and Durrant-Whyte, 2006) or a voting scheme (Olson et al., 2006). Montiel et al. (2006) propose an alternative solution where the system models the inverse of the depth of a landmark rather than the depth; this results in a simpler and more robust algorithm.

Possibly the most versatile but also the most challenging type of SLAM is visual SLAM, using a single hand-held camera. It is versatile because it requires nothing more than a camera which can be hand-held, or attached to a helmet, or fixed on a robot. It is challenging because it must deal with landmark extraction from a high-bandwidth data stream, ambiguous data association, bearing-only measurements, and unpredictable movements. In addition, the memory required to store each individual landmark, the high density of landmarks, and the potentially large size of the map add to the difficulty. Despite these challenges, monocular visual SLAM has been demonstrated to work well (Davison et al., 2007; Clemente et al., 2007; Newcombe and Davison, 2010), although only for maps of limited size.

An essential difference between different types of landmark sensor is whether the landmarks are uniquely identifiable or not. Many types of artificial landmarks, such as ultrasound or radio beacons, sonar transponders, RFID tags, and fiducial markers, are uniquely identifiable. In this case, data association is not a problem and the SLAM system can be simpler and more robust as it does not need to deal with multiple hypotheses and erroneous matches. Naturally occurring landmarks, such as the features detected by visual SLAM systems or laser range finders, are ambiguous, and the corresponding SLAM algorithms must take additional measures to achieve reliable convergence.

Pedestrian SLAM

In principle, the SLAM algorithms we have mentioned from the field of robotics or autonomous vehicles can be applied to pedestrians. Some visual SLAM systems are implemented with pedestrians in mind (Clemente et al., 2007; Strasdat et al., 2010), but other types of sensor are not immediately suitable for use by pedestrians. For instance, laser range finders which detect distance and direction to obstacles are frequently used for robot SLAM where the movement of the robot ensures that the measurements are all in a consistent horizontal plane. This is more difficult to achieve for pedestrians. HeadSLAM (Cinaz and Kenn, 2008b) addresses this by adding an inertial measurement unit to a helmet mounted laser scanner. The inertial measurements enable the system to compensate for the varying tilt of the scans and to perform pedestrian dead-reckoning before applying techniques from robotics. Kleiner and Sun (2007) describe a form of SLAM specifically for pedestrians, using inertial dead-reckoning and RFID landmarks. They construct the map of RFID positions and the paths of the pedestrians offline.

We mentioned a novel application of pedestrian dead-reckoning at the end of the previous chapter. *FootSLAM* uses traces from pedestrian dead-reckoning to construct a map of the walkable areas of the building and does not require any

additional sensors (Robertson et al., 2009). The map consists of a hexagon grid showing which areas are most visited and how they are connected. This reveals the structure of the building, including corridors, rooms, doorways and even furniture. However, it only becomes usable once the pedestrian has covered the main walkable areas of the building several times in different directions.

5.1.2 Contribution

The systems described above are not immediately useful for supporting pedestrian navigation, although the techniques suggested are very relevant. The systems targeted at pedestrians are either unable to provide real-time localisation, or they are not reliable enough for our purposes. FootSLAM is extremely promising but we believe that, in the firefighter navigation scenario, it suffers from the same shortcoming as the pedestrian dead-reckoning on which it is built, namely unpredictable drift. The advantages of FootSLAM, the visualisation of paths and the correction of drift, only become apparent when one or more pedestrians have walked multiple intersecting paths a number of times. In this chapter, we develop and evaluate a SLAM system for pedestrians using inertial pedestrian dead-reckoning and ultrasonic sensor nodes which serve as landmarks. It works in real-time and is particularly suited to pedestrian tracking, including use by emergency responders.

Foot-mounted inertial PDR provides the best possible estimates of a pedestrian's movements and, in principle, works for many types of motion beyond typical walking, including running, crawling, climbing and descending stairs and ladders. The disadvantage is that foot-mounted sensors and their wires are difficult to attach and can get in the way of the wearer. However, since this is a specialised system which would go through several design iterations before being produced commercially, we believe there will be opportunities to address this issue at a later stage, during integration with other elements of firefighting equipment, for instance.

Ultrasonic sensor nodes are an attractive solution for localisation in unprepared environments. They are potentially small and power-efficient enough for response teams to deploy at the scene as required. The nodes provide two-way measurements and distributed computation because they are active devices, contrary to passive RFID tags or visual markers for instance. In some deployments, they can also provide a communication channel between different parts of the building. They are unambiguously identifiable because they broadcast their unique identifier, unlike the visual features used by SLAM systems based on cameras or laser scanners, so data association is not a problem. Unlike radio ranging systems, they take centimetre-resolution range measurements and can be designed to measure bearing as well, and unlike visual markers, they work in low visibility.

We provide an initial proof-of-concept system that we demonstrate in a controlled setting. There are many more aspects to address and we discuss some of these at the end of this chapter. However, we are not aware of any comparable work which has sought to base SLAM on pedestrian dead-reckoning, and we thus believe that this is a valuable step towards a solution for navigation in unprepared environments.

5.2 Implementation: multi-modal sensing and algorithms

In this section, we describe our implementation of a tracking system based on inertial pedestrian dead-reckoning and ultrasonic sensors deployed by the responders as they explore a building. First, we implement an extended Kalman filter (EKF) which can track the pedestrian when the sensor positions are known. Second, we augment this EKF to track the pedestrian and simultaneously locate the sensors in the case where no prior knowledge is available.

5.2.1 Sensing technologies

In these experiments, we use the Kalman filter pedestrian dead reckoning implementation described in chapter 4 and the ultrasonic sensors described in chapter 3. The errors in the range and bearing measurements taken during our previous experiments (ch. 3) are shown in figure 5.1. Each line represents the range or bearing error distributions for one of our eight selected experiments. The true positions of the beacons were surveyed manually, and they were all oriented in the same known direction, along one of the coordinate axes. The ground truth position of the pedestrian was measured using the Ubisense Series 7000 real-time localisation system² based on ultrawideband radio signals. During the experiment, the sensor node attached to the pedestrian was programmed to transmit a series of ultrasonic pulses approximately ten times per second. The nodes on the ground used these pulses to estimate the distance and bearing to the pedestrian.

The large error for the range and bearing measurements in our pedestrian tracking experiment can be explained by three factors. (1) The true position of the pedestrian is measured by the Ubisense ultrawide band localisation system with an accuracy of 15 centimetres according to some documents (Ubisense, 2012); this accuracy is worse than the ranging accuracy of our ultrasonic sensors, and is further limited by a challenging environment in terms of RF propagation. (2) The UWB transmitter used to track the true position of the pedestrian was not co-located with the foot-mounted ultrasonic transmitter but attached to his cap. (3) The ultrasonic transmitter was modified to fit around the pedestrian's shoe but this causes the alignment of the transducers to be sub-optimal; the other foot can also block the line of sight between the transmitter and the sensors lying on the ground. Thus, to give a better indication of our sensors' accuracy, we have also plotted the error from the experiments described in section 3.2. In those experiments, we deployed five static sensors and one mobile sensor on a Lego MindStorms robot in a

²<http://www.ubisense.net> (Accessed 2012.09.24.)

2.75×2.00 m arena. The thick dotted line in figure 5.1 represents the range and bearing errors for over 60 000 measurements involving the mobile sensor in that arena. In these previous experiments, the true sensor positions were measured using visual markers on top of each sensor and camera tracking software which estimates positions with an accuracy of the order of a centimetre, but only over a small area.

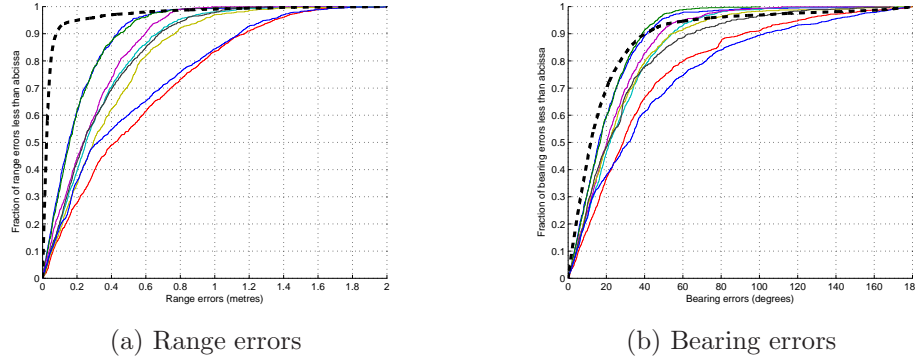


Figure 5.1: Characterisation of the ultrasonic sensors: range and bearing measurement errors. Each of the eight lines represents the error distribution of the unprocessed measurements for a different experiment. The thick dotted line is the error distribution measured in previous work using a more accurate visual fiducial-based tracking system to record the true positions of the mobile sensor.

As we will see in the results presented later in this chapter, the limitations of the ground truth

5.2.2 Localisation and mapping algorithms

We use inertial PDR and our ultrasonic sensor nodes to simultaneously locate a pedestrian and map the positions of the nodes (which effectively serve as landmarks). The simultaneous localisation and mapping (SLAM) techniques which we use have been developed by robotics researchers since the 1980s, but have only been applied to pedestrian localisation in the past decade. We first describe an implementation which assumes that sensor positions are known; then we describe an implementation that also estimates the positions of the sensors, thus requiring no prior knowledge of the environment.

Kalman filtering

The drawbacks of Kalman filter-based SLAM, mentioned in the introduction to this chapter, do not pose a serious problem in our application scenario. First, the sensor nodes we use as landmarks are uniquely and robustly identifiable. Second, the landmarks can be relatively sparse due to the accuracy of the dead-reckoning, and the total time during which tracking is required is limited, thus the total number of landmarks remains small. We also believe this implementation has the potential to be run in real time and distributed over the sensor nodes more easily than the alternatives.

In our application we track a pedestrian. Their state is modeled as $state_p = \begin{pmatrix} x_p & y_p & \psi_p \end{pmatrix}^T$, the two Cartesian coordinates and the direction of travel, or heading. We only address the two-dimensional problem in this work. Tracking the z-dimension is more challenging because our ultrasound sensors only measure bearings in the horizontal plane. During our experiments, our PDR system tended to drift more along the z-axis than in the horizontal plane for some users; however, we later discovered that this was due to insufficient dynamic range on our particular model of inertial sensor and not a fundamental flaw in the algorithm. Tracking in the horizontal plane may be sufficient for many scenarios, including emergency response. If necessary, stairs could be detected and flagged separately.

We have measurements from two sources: PDR which measures the movement of the pedestrian, and ultrasonic sensor nodes which measure the range and bearing to the pedestrian. The PDR measurements are used in the prediction phase, and the ultrasonic measurements are used in the correction phase. The movement model (or process model) takes advantage of the pedestrian dead reckoning measurements $\begin{pmatrix} d & \delta\psi \end{pmatrix}^T$ which tell us how the pedestrian has moved since the previous estimate. d is the distance traveled and $\delta\psi$ is the change in the direction of travel (heading) since the previous PDR measurement. We measure the change in heading because

we know that the absolute heading measured inertially is prone to drift, whereas the short term variations are more accurate. These notations are presented in equation 5.1 and illustrated in figure 5.2. k is the timestep.

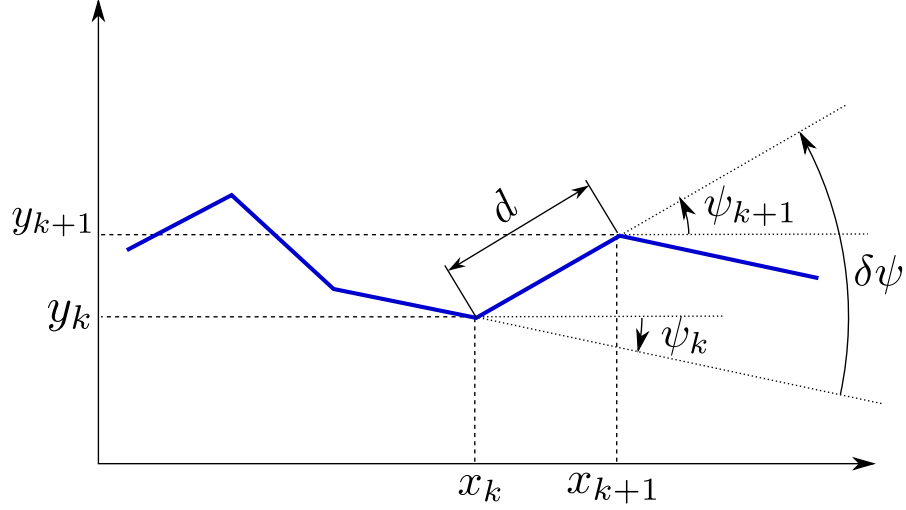


Figure 5.2: Notations used for PDR measurements. d is the distance travelled, $\delta\psi$ is the change in heading since the previous PDR measurement, and k is the timestep.

$$\begin{aligned}
 state_p^{k+1} &= f(state_p^k, meas_{pdr}) \\
 &= \begin{pmatrix} x_{k+1} \\ y_{k+1} \\ \psi_{k+1} \end{pmatrix} \\
 &= \begin{pmatrix} x_k + d \cos(\psi_k + \delta\psi) \\ y_k + d \sin(\psi_k + \delta\psi) \\ \psi_k + \delta\psi \end{pmatrix}
 \end{aligned} \tag{5.1}$$

We assume that both elements of the PDR measurement are subject to additive Gaussian noise. Measurement noise has been omitted from process equation 5.1 where each occurrence of d is actually $d + d \varepsilon_d$, and each occurrence of $\delta\psi$ is actually $\delta\psi + d \varepsilon_{\delta\psi}$ so that the noise is applied proportionally to the distance moved since the previous measurement. The covariance of the noise $\begin{pmatrix} \varepsilon_d & \varepsilon_{\delta\psi} \end{pmatrix}^T$ is denoted Q . By defining the process noise proportional to the distance travelled we prevent

the covariance (the position uncertainty) from increasing when the pedestrian is standing still.

The ultrasonic sensor nodes are able to measure the range r and the bearing ϕ to a compatible device attached to the pedestrian's boot and colocated (as closely as possible) with the inertial measurement unit used to perform PDR. The state of each sensor i is modeled as $state_{s_i} = \begin{pmatrix} x_{s_i} & y_{s_i} & \theta_{s_i} \end{pmatrix}^T$, the two Cartesian coordinates and the orientation. We can express the measurement as a function of the state of the pedestrian and of the relevant sensor, as shown in equation 5.2. For an illustration of the measurement model, see figure 3.4 in chapter 3.

$$\begin{aligned}
 predicted_meas_{us} &= h(state_p, state_{s_i}) \\
 &= \begin{pmatrix} r \\ \phi \end{pmatrix} \\
 &= \begin{pmatrix} \sqrt{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2} \\ \arctan\left(\frac{y_p - y_{s_i}}{x_p - x_{s_i}}\right) - \theta_{s_i} \end{pmatrix}
 \end{aligned} \tag{5.2}$$

We assume that the ultrasonic measurements are subject to additive Gaussian noise $\begin{pmatrix} \varepsilon_r & \varepsilon_\phi \end{pmatrix}^T$ with covariance R . This has been omitted from measurement equation 5.2 for simplicity. It is important to ensure that all angle differences remain between $-\pi$ and π so the filter can determine whether two angles are similar or not.

The process function f (eq. 5.1) and the measurement function h (eq. 5.2) are both non-linear functions of the state, measurements, and noise, so we need to calculate their Jacobian matrices (eq. 5.3, 5.4 and 5.5). The Jacobian $W = \frac{\partial h}{\partial \varepsilon_{us}}$ is the identity matrix because additive noise is applied directly to the measurement,

it can therefore be omitted from the expression of the gain K in equation 5.7.

$$\begin{aligned}
 A &= \frac{\partial f}{\partial state_p} \\
 &= \begin{pmatrix} 1 & 0 & -d \sin(\psi_p) \\ 0 & 1 & d \cos(\psi_p) \\ 0 & 0 & 1 \end{pmatrix}
 \end{aligned} \tag{5.3}$$

$$\begin{aligned}
 H &= \frac{\partial h}{\partial state_p} \\
 &= \begin{pmatrix} \frac{x_p - x_{s_i}}{\sqrt{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2}} & \frac{y_p - y_{s_i}}{\sqrt{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2}} & 0 \\ -\frac{y_p - y_{s_i}}{(y_p - y_{s_i})^2 + (x_p - x_{s_i})^2} & \frac{x_p - x_{s_i}}{(y_p - y_{s_i})^2 + (x_p - x_{s_i})^2} & 0 \end{pmatrix}
 \end{aligned} \tag{5.4}$$

$$\begin{aligned}
 V &= \frac{\partial f}{\partial \varepsilon_{pdr}} \\
 &= \begin{pmatrix} d \cos(\psi_p) & -d^2 \sin(\psi_p) \\ d \sin(\psi_p) & d^2 \cos(\psi_p) \\ 0 & d \end{pmatrix}
 \end{aligned} \tag{5.5}$$

We can apply the Kalman prediction equations whenever a PDR measurement is available (eq. 5.6).

$$\begin{aligned}
 predicted_state_p &= f(state_p, measurement_{pdr}) \\
 predicted_P_p &= AP_p A^T + VQV^T
 \end{aligned} \tag{5.6}$$

Similarly, when an ultrasonic measurement is available we apply the Kalman update equations (eq. 5.7).

$$\begin{aligned}
 K &= PH^T (HPH^T + R)^{-1} \\
 updated_state_p^{k+1} &= predicted_state_p^k + K (meas_{us} - predicted_meas_{us}) \\
 updated_P_p^{k+1} &= (I - KH) predicted_P_p^k
 \end{aligned} \tag{5.7}$$

Pedestrian SLAM

If the positions of the sensor nodes are unknown, then we have a simultaneous localisation and mapping (SLAM) problem where we must estimate the positions of the sensors (the map) in addition to the position of the pedestrian. This can be achieved by modifying the Kalman filter developed above. The positions of the sensors become variables instead of constants and the filter also tracks their covariances.

The prediction equations remain the same – the filter uses the PDR measurements to predict the position of the pedestrian. The sensors are assumed to be static so their position estimates remain the same during the prediction phase. However, the correction phase is slightly different because each ultrasonic measurement can correct both the pedestrian position estimate and the position estimate of the sensor which took the measurement.

The state under consideration is now the concatenation of the position of the pedestrian and the position of the sensor i which took the measurement (eq. 5.8). The Jacobian matrix H is also different because now the elements of the sensor state $state_{s_i}$ are also variable (eq. 5.9).

$$\begin{aligned} state_{p+s_i} &= \begin{pmatrix} state_p \\ state_{s_i} \end{pmatrix} \\ P_{p+s_i} &= \begin{pmatrix} P_p & 0 \\ 0 & P_{s_i} \end{pmatrix} \end{aligned} \tag{5.8}$$

$$\begin{aligned}
H &= \frac{\partial h}{\partial \text{state}_{p+s_i}} \\
&= \begin{pmatrix} \frac{x_p - x_{s_i}}{\sqrt{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2}} & -\frac{y_p - y_{s_i}}{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2} \\ \frac{y_p - y_{s_i}}{\sqrt{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2}} & \frac{x_p - x_{s_i}}{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2} \\ 0 & 0 \\ -\frac{x_p - x_{s_i}}{\sqrt{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2}} & \frac{y_p - y_{s_i}}{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2} \\ -\frac{y_p - y_{s_i}}{\sqrt{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2}} & -\frac{x_p - x_{s_i}}{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2} \\ 0 & -1 \end{pmatrix}^T \quad (5.9)
\end{aligned}$$

The combined state state_{p+s_i} is created when each ultrasonic measurement is received, and when the correction phase is finished the position of the pedestrian and the position of the sensor are stored separately. This allows us to only process data of dimension six for each measurement. This scales better and is computationally much less expensive than considering all sensors when doing the update. However, there is a tradeoff because we lose the information about the cross-correlations between the different sensor positions which would allow each ultrasonic measurement to correct position estimates for other nodes as well as the one that took the measurement.

In a typical Kalman filter the covariance increases during the prediction phase due to the uncertainty of the process model and then decreases during the correction phase due to the information from the measurement.³ However, the covariances of the sensors never increase because they are modeled as static. To avoid the covariances decreasing too quickly we use the fading memory technique described by Simon (2006) which artificially increases the sensor covariances before each correction phase by multiplying P_{s_i} by a value slightly greater than one (Simon (2006) suggests 1.01^2 and we also use this value). Without this, the sensor covariances would converge to zero and their position estimates would never be updated.

³There are exceptions, notably following a 180 degree turn when the covariance can decrease for a brief period even during the prediction phase.

Sensor initialisation

The first time a sensor reports a measurement, the filter has no information about it. A sensor cannot be initialised from a single measurement, so we delay initialisation until we have at least three measurements taken from positions that are well spaced apart. This is not a limitation in practice because we only need to know the position of the sensor when we return to the same location later in the mission, not immediately after dropping it. Using these measurements, we can usually perform reliable trilateration using a non-linear regression. However, if the points are nearly collinear there are two solutions. The bearing measurements help us resolve such ambiguities, but we find that using them directly in the regression gives poor results. We adopt the following heuristic method for selecting the correct solution. First, we use only the range measurements in the regression. We know that the solution computed by the regression is either the true position of the sensor, or a position that is the symmetric of the true position with respect to the path travelled. Therefore, we estimate the least squares line which approximates the path travelled, and compute the symmetric of the estimated sensor position. One of these points should be close to the true sensor position. In other words, if there is indeed an ambiguity we are likely to have found two local minima corresponding to the actual position of the sensor and its reflection. Using each of the bearing measurements, we estimate the orientation of the sensor for both positions. The correct position will yield similar orientation estimates for all of the bearing measurements. The incorrect symmetric position will yield inconsistent orientation estimates for each of the bearing measurements. We select the position which minimises the bearing residuals (i.e., the variance of the orientation estimates).

Figure 5.3 illustrates the sensor initialisation process. In this example, the pedestrian walked along a straight path for approximately two metres. An ultrasound sensor took range and bearing measurements to the pedestrian and we recorded those that were at least 50 centimetres apart. The circles represent the range mea-

surements taken at these different points, and their intersections are the possible positions of the sensor. There are two solutions because the pedestrian path was almost linear. Using a single bearing measurement is not enough to choose the correct solution, but the sequence of bearing measurements is only consistent with the correct solution. This heuristic initialisation method works well but, because of the noise in our bearing measurements, there are still times when it is unable to choose correctly between the true sensor position and the symmetric.

Djugash et al. (2006) use a similar method for initialising sensor positions in their work on range-only SLAM for robots, but they explain it differently (and do not have the benefit of bearing measurements). They represent the sensors as nodes in a graph, and robot positions as *virtual nodes*. The edges are either sensor-to-sensor ranges (which we do not use) or sensor-to-robot ranges. Based on this graph representation, they are able to determine the locations of sensor nodes and of the robot by running a batch optimisation. Deans and Hebert (2000) also use a comparable approach in their Kalman filter SLAM implementation for bearing-only measurements. They initialise a landmark position by performing *bundle adjustment* (a non-linear optimisation) over a section of the robot trajectory and the first few bearing measurements in order to optimise both the landmark position and the robot trajectory (we only optimise the landmark positions).

5.3 System evaluation

We evaluate the EKF algorithm (with and without SLAM) described in the previous section using multiple sets of inertial and ultrasound measurements. These were recorded as a subject walked through an indoor office environment. In this section, we first describe the experimental setup, then we show the results achieved for different experiments, before finally drawing conclusions about the suitability of this type of algorithm for emergency response scenarios.

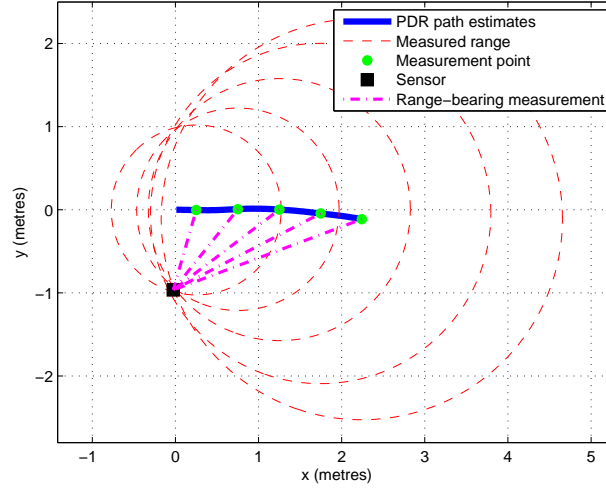


Figure 5.3: Sensor initialisation from a sequence of range/bearing measurements. First the ranges are used to determine the possible sensor locations. Then the bearings are used to select the correct one.

5.3.1 Description of experiments

Our experiments took place in an office building at the University of Twente in the Netherlands. The test area covered a large, mostly empty, office on one side of a corridor, and two smaller individual offices, each with a desk, on the other side of the corridor (figure 5.4).

Inertial and ultrasound data collection

We evaluate our algorithms using data gathered during eight different collection sessions, each lasting from three to eight minutes. We placed twenty-one ultrasound sensors on the floor (fig. 5.5(a)), and surveyed their positions by hand. Using a total station⁴, we also surveyed six additional reference points which the pedestrian walked to and from in some of the experiments.

During the data collection sessions, a pedestrian (the author) walked within the test area at an average walking speed between 0.8 and 1.5 metres per second.

⁴A total station is a precision surveying tool similar to a theodolite.

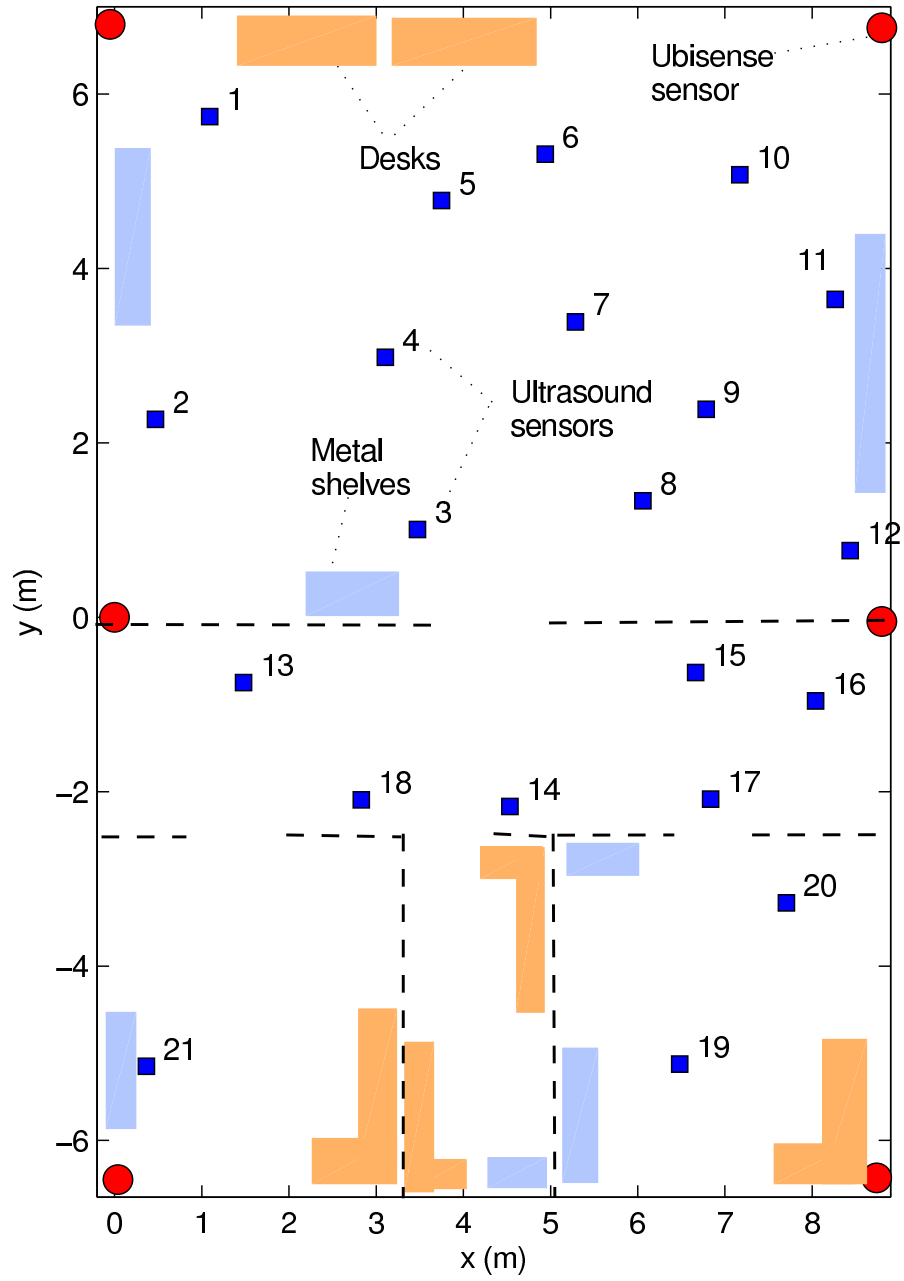


Figure 5.4: Test area: a large office, corridor, two smaller offices covering a total of 9×13 metres.

The pedestrian wore an XSens MTx inertial sensor on his foot, and an ultrasonic transmitter on his lower leg (fig. 5.5(b)). The transducers of the transmitter were on flying leads and attached around the edge of the sole so they would be in the same plane as the sensors deployed on the ground. The paths are shown in figure 5.8 in the results and analysis section. Paths (a), (g) and (h) were constrained to the large room. All other paths covered the large room, both smaller rooms, and the corridor. For path (a), the pedestrian walked around the room once and then to reference point 1 (not shown), then around the room again and to reference point 2, and so on. For paths (b), (c), (d), and (e), he walked continuously between the rooms following the same path each time; the path in the large room is a straight line across the middle of the room, hence the ‘T’ shape. Path (f) is similar but he walked around the large room instead of across it. For paths (g) and (h), the pedestrian stopped at each reference point (not shown) for approximately ten seconds after walking arbitrarily about the room for approximately a minute between each reference point. These different experiments contrast situations where the same path is repeated multiple times with situations where the movement is more varied. The scenario where the pedestrian stops at known positions would potentially allow us to evaluate the accuracy of our position estimates even in the absence of dynamic ground truth.

Path capture using Ubisense

Ground truth was measured using a commercial Ubisense Series 7000 ultrawide band (UWB) localisation system⁵. We placed six Ubisense receivers at the edges of the measurement volume. This system was calibrated using the Ubisense software’s “full calibration” mode (the most accurate but time consuming option). The pedestrian wore a Ubisense tag on a cap in order to measure reference location data. Ideally, the tag would have been colocated with the other sensors on the foot, however, we tried this and found that the body shields the tag from the receivers.

⁵<http://www.ubisense.net> (Accessed 2012.09.24.)



(a) Sensor nodes deployed along the corridor.



(b) Ultrasound transmitter (blue) and Xsens inertial measurement unit (orange) attached to the pedestrian's foot; transducers around the edge of the sole.



(c) Data collection performed in the large office. Ultrasound sensors are deployed on the floor.

Figure 5.5: Experiment and data collection.

However, from our previous work, we know that the ninetyeth percentile horizontal accuracy of this technology can vary from 11 cm to worse than 150 cm (Muthukrishnan and Hazas, 2009, fig. 4 and tab. 2), depending on calibration effort invested, sensor density, and environmental conditions affecting UWB line-of-sight and multipath. Thus, the Ubisense accuracy can be considerably worse than the ultrasonic sensor ranging accuracy, which we measured in a separate set of experiments to be about 7 cm at the ninetyeth percentile confidence level (fig. 5.1). It therefore cannot be used to characterise the lower bound of the accuracy for our pedestrian tracking and SLAM algorithms. However, it does give a reliable indication of the path traversed, and can to some extent be used to compare the relative accuracy of different types of path, side-by-side (assuming both types of path cover similar parts of the measurement volume, and thus will be subject to similar amounts of Ubisense tracking error). Finally, unlike the error in the PDR position estimates, the absolute error of the Ubisense system is bounded making it useful for ground truth.

Error characterisation

Typically, a localisation system is characterised by the error between the estimated position of the pedestrian and his true position, and the estimated positions of the sensors and their true positions. For our system this is not possible because the SLAM algorithm works in its own arbitrary coordinate system which can potentially change over time. In other words, the positions of the pedestrian and of the sensors can only be compared to each other, not to their true values. For this reason, we calculate the errors in the ranges and bearings between sensors, and between sensors and pedestrian. We can then examine how these errors evolve over time and compare their distributions. This is very similar to how we evaluated our sensor node localisation algorithm in chapter 3. The performance of the filter can also be evaluated to some degree without any ground truth, by examining the

innovations (i.e., the differences between the predicted ranges and bearings, and the actual measurements). If the innovation remains small, then the filter is likely to have correctly estimated the positions of the pedestrian and of the sensor.

The estimated positions of the sensors may be aligned with their hand-surveyed positions either manually or by using a regression to determine the optimal translation and rotation to apply. This method can be useful for visualising the output of the filter but it must be used with care because of the additional complexity introduced by the optimisation and the fact that the coordinate system in which the filter locates the sensors and the pedestrian changes over time.

Algorithm parameters and SLAM initialisation

The specific algorithm parameters we use are given in table 5.1. Note that the values we use for Q and R bear no clear relationship to measurable noise values, and were chosen empirically. This is a common situation when designing a Kalman filter for which the system model is not well known or not detailed enough; the modeled noise values need to be increased in order to account for the modeling “errors”. Measurements for which the range innovation divided by the square root of the sum of x and y covariances for the pedestrian and the relevant sensor are greater than 0.5 are discarded as outliers. In other words, we discard measurements that are very different from what we expected, unless we are very unsure of our current estimates.

In the case where we assume that the sensor positions are known in advance, we initialise them to their true values and set P_{s_i} to zero. The initial position of the pedestrian is set according to the Ubisense position estimates. Their direction of travel is initialised by using the angular difference between the direction given by the first two Ubisense measurements that are at least three metres from each other, and the corresponding position estimates using PDR alone. In the case where we

Table 5.1: Parameters used in the SLAM algorithm.

Name	Notation	Value
PDR noise covariance	Q	$\begin{pmatrix} 2 & 0 \\ 0 & 20 \end{pmatrix}^2$
Range/bearing measurement noise covariance	R	$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}^2$
Initial sensor position covariance	$P_{s_i}^0$	$\begin{pmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 8.73 \end{pmatrix}^2$
Initial pedestrian position covariance	P_p^0	(0)
Fading memory factor		1.01^2

assume no prior knowledge of sensor positions, we use five ultrasonic measurements taken at least 30 cm from each other to initialise the sensor positions, as explained earlier.

5.3.2 Results and analysis

We now take a closer look at the different components of the system and how they perform on our datasets.

Inertial Pedestrian Dead Reckoning (PDR)

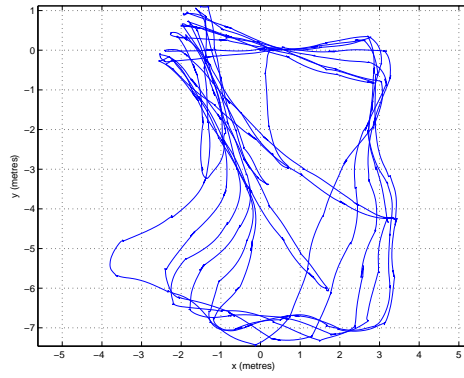
The inertial pedestrian dead reckoning measurements can be used for determining the path, but, as explained in chapter 4, the estimated position drifts over the course of the experiment. Figure 5.6 shows how the position estimates suffer from drift when the PDR measurements are used alone. In path (b), all iterations are superimposed so we can assume that there is very little error in the position estimates. Paths (c), (d), (e), and (f) display slight rotational and translational drift of up to ten degrees and three metres respectively. Path (a) shows slightly more drift but the roughly rectangular shape of the path remains clear throughout the

experiment. Paths (g) and (h) are difficult to analyse at this stage because there is no repeating pattern, but we will see that this is beneficial when we attempt to locate the sensor nodes.

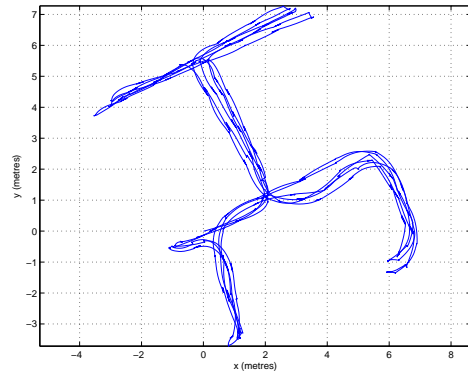
We know from observing these traces that the uncertainty in the PDR position estimates increases with distance travelled. The process model in our Kalman filter is designed to take this into account and the value of the pedestrian position estimate covariance should therefore increase with distance travelled (and with time). Figure 5.7 shows how the estimated position error covariance changes with time for experiment (d). We plot $\sqrt{|P_{p,xy}|}$ (square root of the determinant of the estimated position error covariance) versus time which is an indicator of the pedestrian position uncertainty, and $\sqrt{P_{p,\psi}}$ (square root of the heading estimate error covariance), which is an indicator of the pedestrian heading uncertainty. The uncertainty starts at zero because we arbitrarily decide on the initial position and heading of the pedestrian. We see that globally the uncertainty increases as expected but there are several instances where the uncertainty decreases. This occurs on corners when the change in heading is large. This decrease in uncertainty is normal, as position errors can cancel each other out for some time following a 180 degree turn. Intuitively, a pedestrian will become somewhat “less lost” if they turn around and walk back towards their starting point for a short time. Wan and Foxlin (2010) give a more detailed analysis of PDR error, including this phenomenon.

Kalman filter using known sensor positions

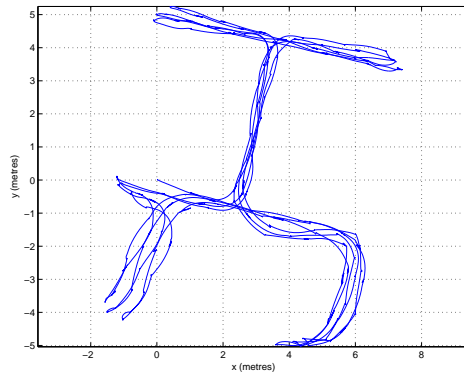
If the positions and orientations of the sensors are known in advance, we can use the first form of our Kalman filter to correct the PDR location estimates. In our implementation of the Kalman filter, we set the sensor and pedestrian positions and orientations based on the surveyed positions. Setting the estimated sensor position error covariances to zero ensures that their position estimates do not



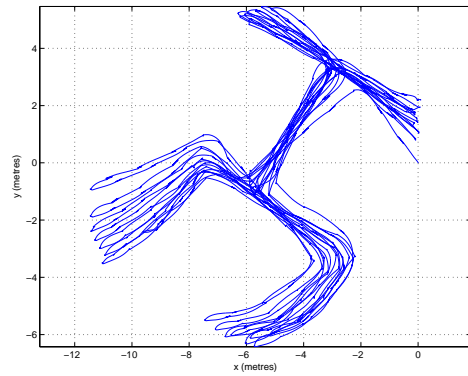
(a) Large room



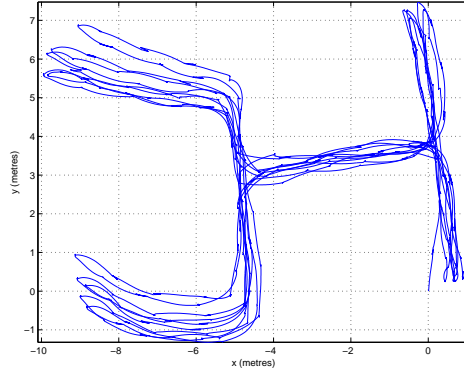
(b) All rooms (T) 1



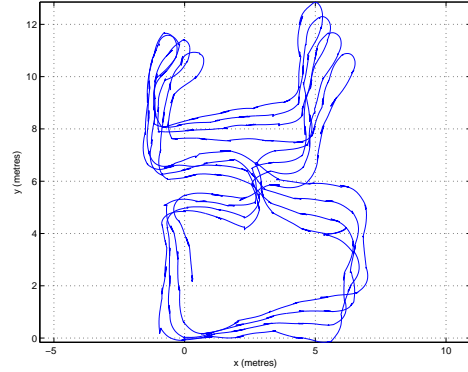
(c) All rooms (T) 2



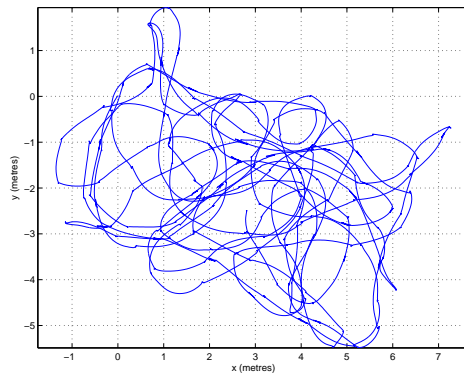
(d) All rooms (T) 3



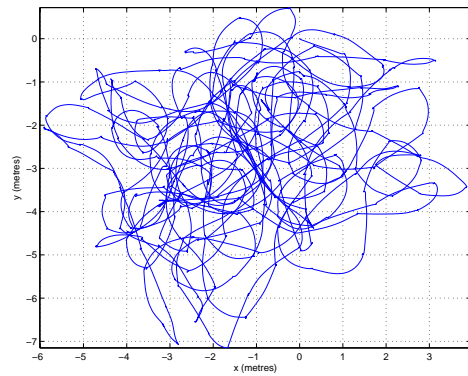
(e) All rooms (T) 4



(f) All rooms (O)



(g) Random 1



(h) Random 2

Figure 5.6: Paths estimated from inertial pedestrian dead reckoning alone.

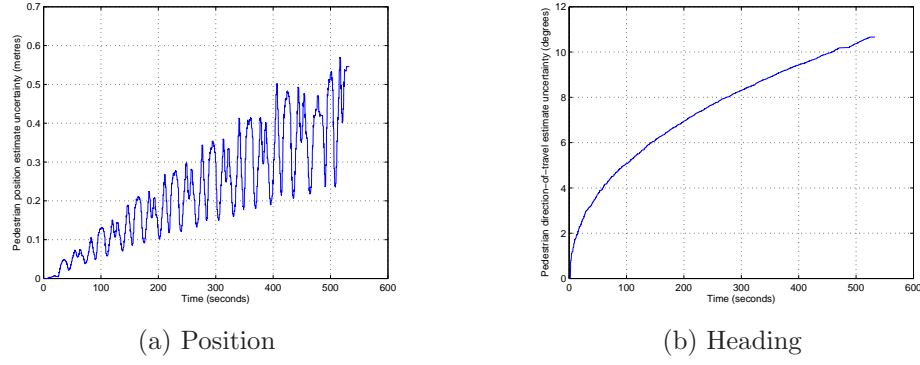


Figure 5.7: Change in the uncertainty of the pedestrian position and heading estimates when PDR is used alone (with ZUPTs and orientation estimation). Overall the uncertainty in position increases linearly over time (the covariance is quadratic), and the uncertainty in heading increases with the square root of time (the covariance is linear). Created from experiment (d).

change as range and bearing measurements are processed by the filter, only the pedestrian position is affected.

The results are shown in figure 5.8. Much of the drift observed in the PDR-only traces (fig. 5.6) has been corrected. Experiments (a) to (f) all show similar errors near the corners of the arena. There is an offset of approximately one metre between the ground truth positions and the estimated positions. We note that the estimated positions are all consistent with each other (i.e., they are all aligned). Therefore, we suspect that these errors are due to inaccuracies of the Ubisense location estimates which can worsen when the tracked tag is placed close to the limits of the covered area. As previously, experiments (g) and (h) are difficult to evaluate, however, we observe that the estimated position of the pedestrian remains correctly constrained to the upper part of the area, and that several sections of the ground truth path can be matched to corresponding parts of the estimated path.

Figure 5.9 illustrates how the covariance of the pedestrian position estimate evolves during these experiments when range and bearing measurements are taken into account. As they start to walk, the covariance increases due to the potential error in the PDR estimates, but when ultrasound measurements are received the

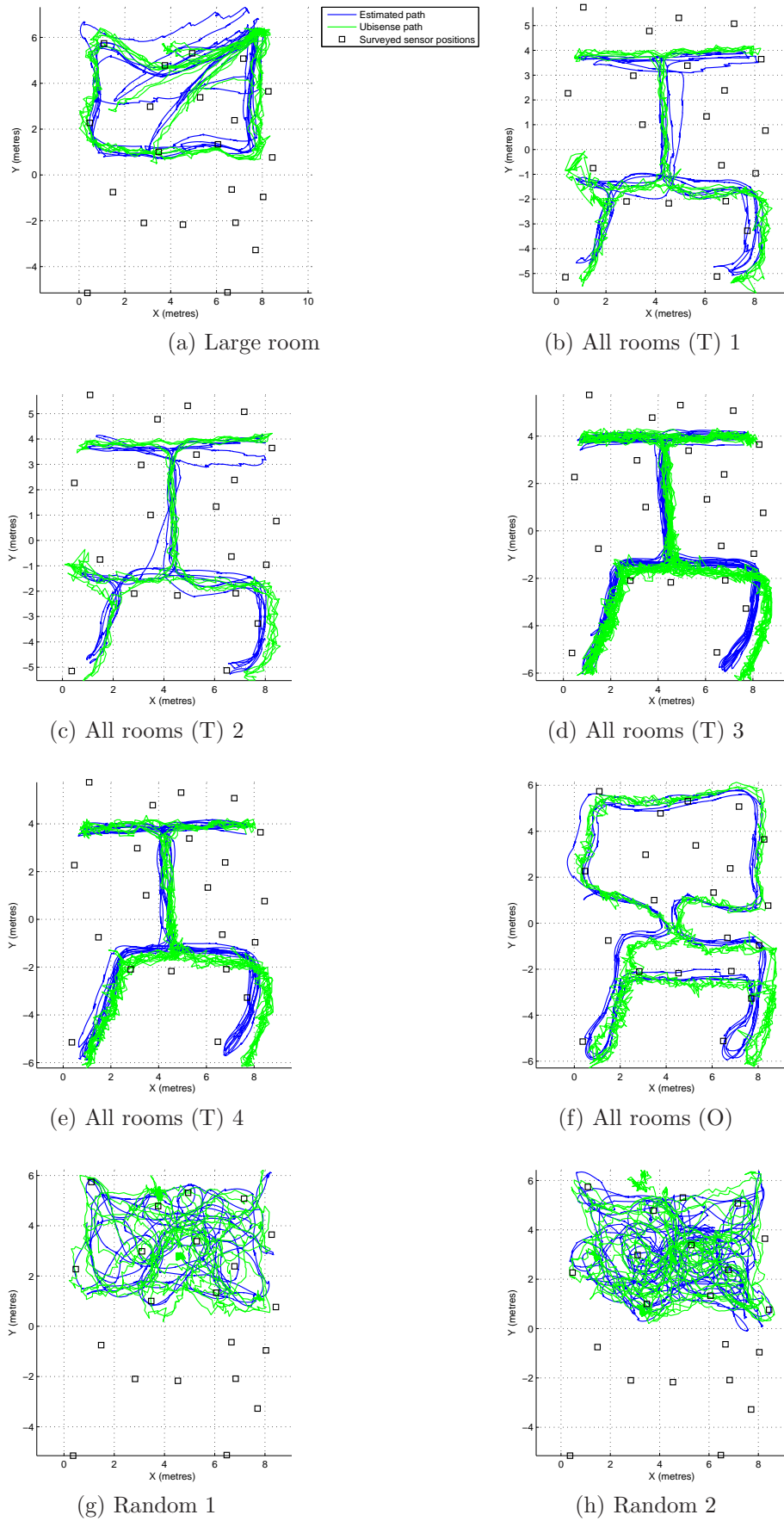


Figure 5.8: Paths estimated from inertial pedestrian dead reckoning and ultrasonic measurements from sensors with known positions. Our path estimates are in blue; the Ubisense ground truth is in green. The sensors are shown as black squares.

covariance decreases due to the additional information. Eventually, a regular cycle of increases due to PDR and decreases due to ultrasonic measurements maintains the covariance around a constant value. As mentioned earlier, the exact values of the covariance (or its square root) do not always map well to true errors, due in particular to the approximations and non-linearities in our model.

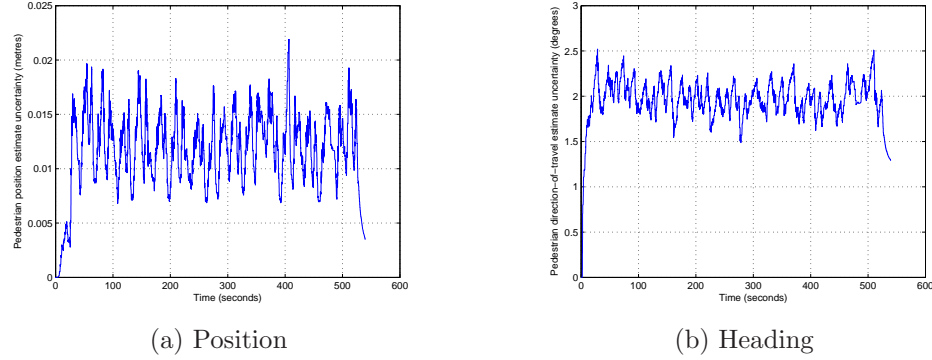


Figure 5.9: Change in the uncertainty of the pedestrian position and heading estimates when PDR is combined with ultrasonic measurements from landmarks with known positions. Uncertainty increases over time due to PDR error accumulation, but decreases when ultrasonic measurements from sensors are available. Created from experiment (d).

Simultaneous Localisation And Mapping (SLAM)

Visualising the output of the SLAM filter as we did when the sensor positions were known in advance can be misleading because the estimated coordinates of the sensors change during the course of the experiment as they are updated by the SLAM process. Figure 5.10 shows that in all the experiments, the estimated positions of the sensors change over time. Note that these plots are in an arbitrary coordinate system and do not directly map to the surveyed sensor positions without first finding and applying the most appropriate rotation and translation. Initially, they move a lot as more measurements are taken but, even after they have stabilised, they continue to drift slowly. Since our *map* is defined by the sensor positions, it also moves. In other words, this SLAM filter only gives positions of the sensors and of the pedestrian relative to the other sensors, not in an absolute coordinate system.

This means that the filter might provide different coordinates for the pedestrian after he returns to a previous location, but this could still be correct (in a relative sense) if the estimated positions of the sensors have also changed during that time. Conversely, if the estimated position of the pedestrian remains the same as previously but the estimated coordinates of the sensors have changed the result could be incorrect (in a relative sense). This is because we are performing online SLAM. Offline methods such as GraphSLAM (Thrun and Montemerlo, 2006), which optimise the complete trajectory estimate and map after all data has been recorded, do not suffer from this limitation. They can be used to display the complete path and the landmarks on a single map.

In our experiments, we find that we are able to display the estimated path of the user (fig. 5.11) despite the potential issues described above. The drift in sensor positions (fig. 5.10) may have been small enough to not interfere with the visualisation. Figure 5.11 also shows the estimated positions of the sensors at the end of each experiment. These estimates were aligned to the surveyed sensor positions after running a non-linear regression to determine the affine transformation (rotation and translation) that minimises the sum of squared errors between surveyed and estimated positions. The green line represents the Ubisense estimated path which is already in the same coordinate space as the surveyed sensor positions.

The path estimates are similar to the ones obtained with prior knowledge of the sensor positions, and most of the sensors are placed within 30 centimetres of their true positions. In experiments (c) and (e), there are a number of sensor nodes with incorrectly estimated positions (although the path estimates are accurate). This is a consequence of the ambiguities in the initialisation method – each of these sensors is placed on the wrong side of the path. As explained earlier in the chapter, this occurs due to the combination of two factors – (1) the measurements used for the initialisation of the sensor are taken from points which are nearly collinear, (2) the bearing measurements are too noisy to determine which of the two possible positions is correct, so our heuristic selection method (fig. 5.3) fails. The consequences

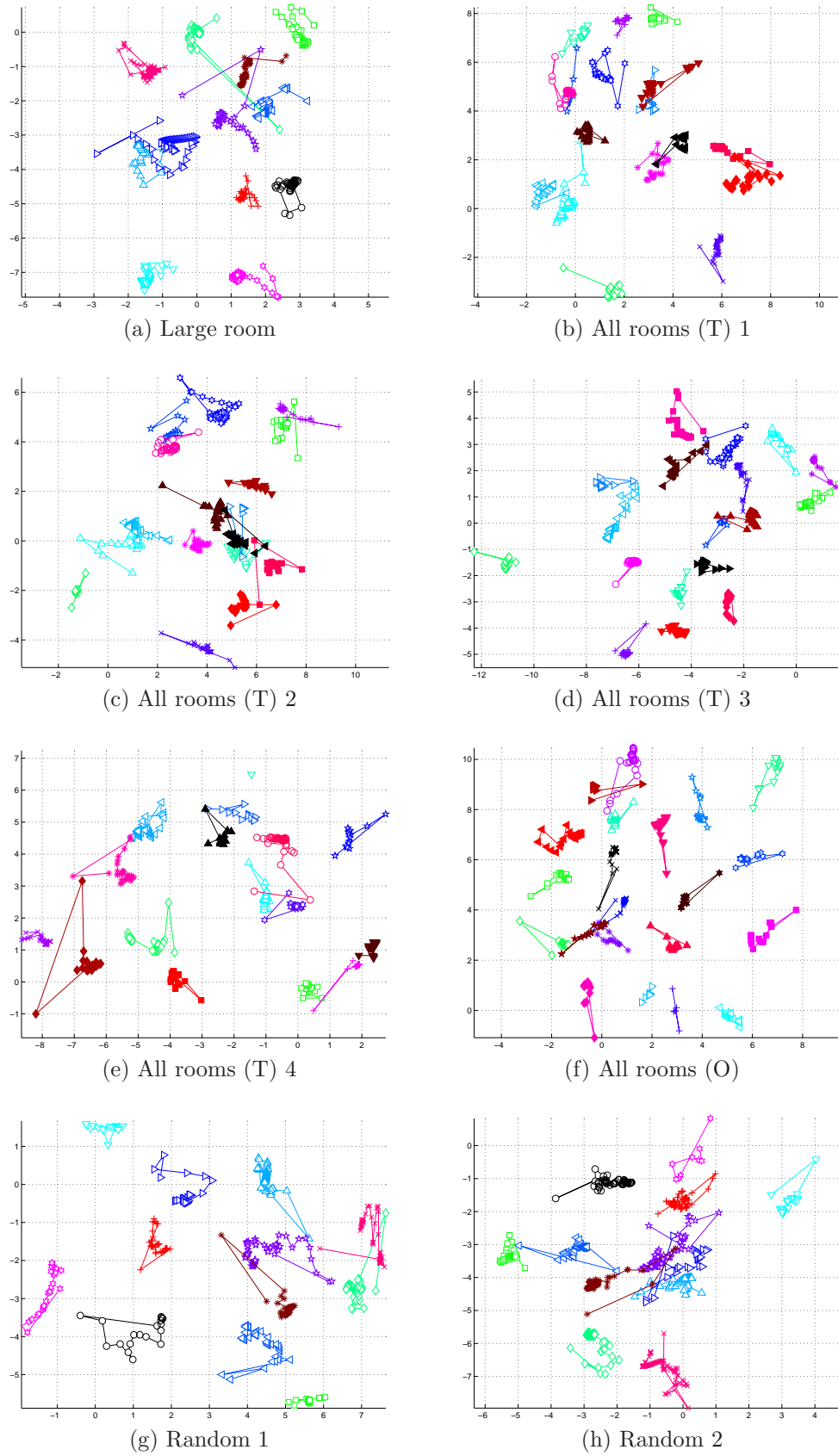


Figure 5.10: Changes in the estimated positions of the sensors during the experiments. Initially, the estimates change a lot, then they stabilise while continuing to drift slowly. Each sensor is shown in a different colour.

can be quite small if the misplaced sensor is only in range of the straight section of path with which it was initialised, for instance the two sensors in the upper right and left corners of figure 5.11(e). But in general, these types of errors in the sensor position estimates will create error in the pedestrian position estimates. This problem occurs for our data because the sensors were deployed in advance (although their positions were not made available to the SLAM algorithm), an unfortunate artefact of our experimental setup. In a practical scenario, the sensors would be dropped or placed near the feet of the pedestrian as they walked around. Their positions could be directly initialised with the pedestrian's current position estimate, thus eliminating any ambiguity. Their orientations could be initialised using subsequent bearing measurements. This is the method we adopt in the following chapter.

Figures 5.11(g) and 5.11(h) provide particularly good estimates of the sensor positions. This is probably because these paths did not include any straight sections, therefore the initialisation was less likely to be ambiguous and sensors that were initialised incorrectly were adjusted thanks to the variety of range/bearing measurements taken from many different positions. In other words, the SLAM solution benefits from favourable geometric dilution of precision. This bears some similarity to situations where planes or ships are required to perform a particular manoeuvre in order to improve their tracking of a target by increasing the observability of its position (Song, 1999).

In order to evaluate the performance of the filter in a more quantitative manner, we look at the range and bearing errors between the sensors, and between the sensors and the pedestrian for every update (figs. 5.12 and 5.13). These errors reflect how accurately the sensors and the pedestrian are positioned relative to each other. As expected, in most cases the errors for SLAM are higher than when the sensor positions are known a priori. In almost all cases the nintieth percentile range error between the pedestrian and the sensors is less than two metres. This value reflects how well the pedestrian can be located in the map.

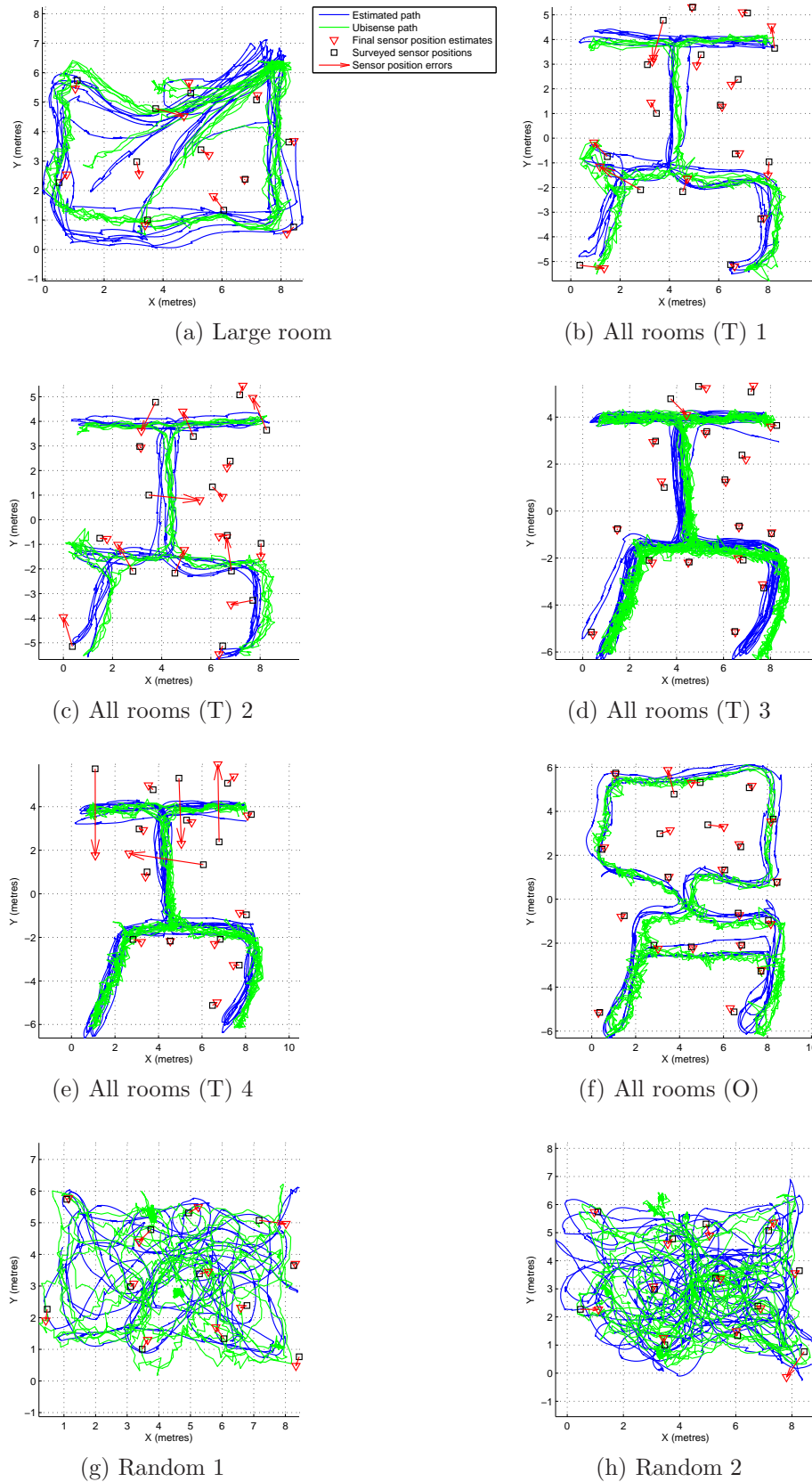


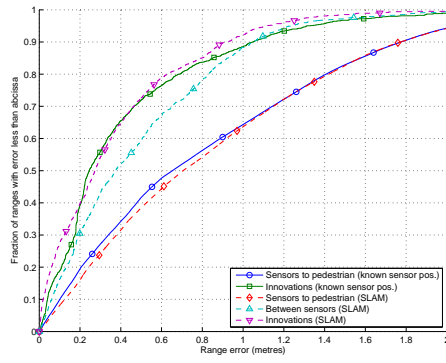
Figure 5.11: SLAM performance: estimated pedestrian path and estimated positions of sensors at the end of each experiment rotated and translated to minimise the distance with the surveyed positions. Our estimated path is in blue; the Ubisense ground truth is in green. Our final sensor estimates are red triangles; the true sensor positions are black squares.

Figures 5.12 and 5.13 also show the innovations, i.e., the differences between the predicted measurements and the actual measurements. The range innovations tend to be smaller than the corresponding estimated errors. This again suggests that using the position estimates from the Ubisense localisation system as groundtruth overestimates some of the errors.

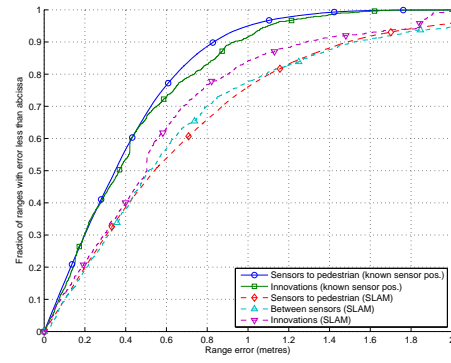
The bearing errors from figure 5.13 are more difficult to interpret because they depend on the position error of the pedestrian and sensors, and on the orientation error of the sensors. For instance, if a sensor's orientation is correct but the pedestrian is estimated to be a few centimetres in front of it, instead of a few centimetres behind it, the bearing error could be 180 degrees.

In figures 5.12 and 5.13, we show errors between all sensors, but it would also be reasonable to only take into account sensors that are either close to or far from the pedestrian depending on whether we are interested in local accuracy (position relative to nearby sensors, necessary for navigation) or global accuracy (position relative to sensors which are far away, necessary for route planning). In figures 5.14 and 5.15, we have plotted separately the errors between sensors, and between pedestrian and sensors when they are less than three metres apart, and those errors when they are more than three metres apart. The three metre limit is arbitrary, but corresponds to an area which could quickly be searched by a firefighter equipped with a long-handled tool. These figures show that in many cases the local range error for SLAM is close to the error when the sensor positions are known. When the far range errors are larger than the local errors, this is due to large scale distortion of the sensor positions. Large scale distortion makes it difficult to overlay the estimated sensor positions onto a map or floorplan, but should not affect indoor navigation scenarios where a firefighter uses only nearby sensors as landmarks to progress towards a target in small steps.

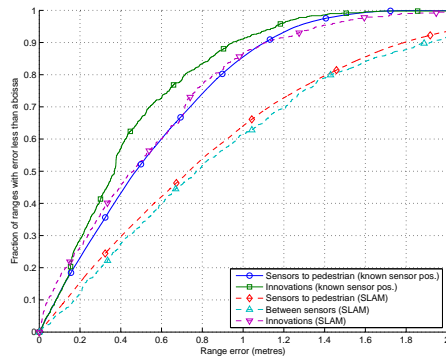
In the case where sensor positions are known in advance, the bearing errors from far away sensors tend to be much smaller than the local bearing errors. Due to



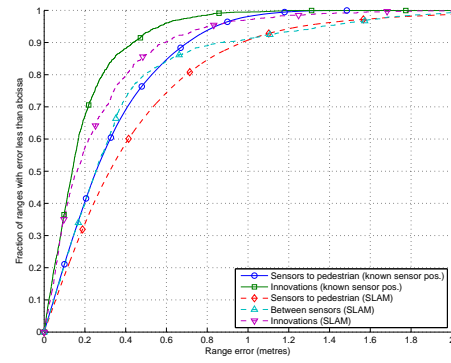
(a) Large room



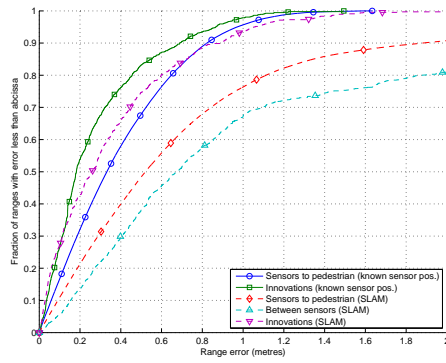
(b) All rooms (T) 1



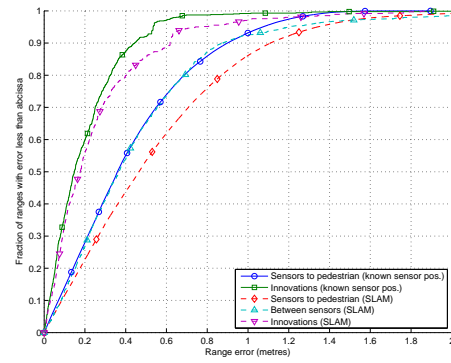
(c) All rooms (T) 2



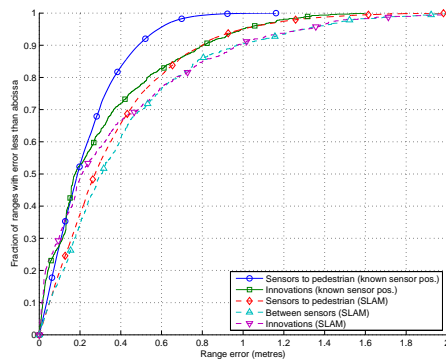
(d) All rooms (T) 3



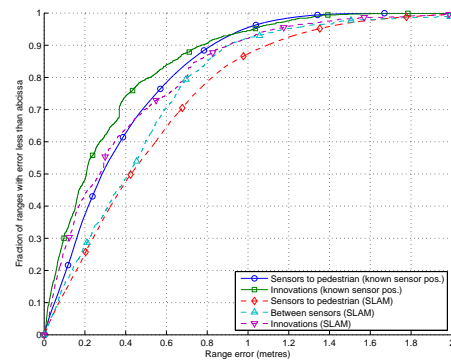
(e) All rooms (T) 4



(f) All rooms (O)

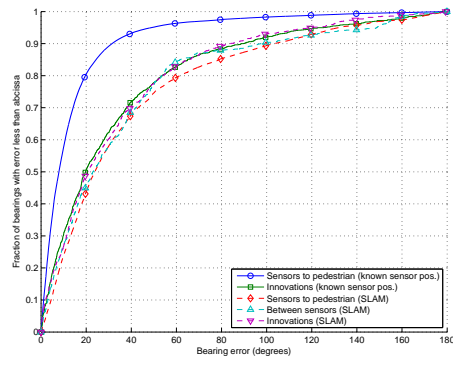


(g) Random 1

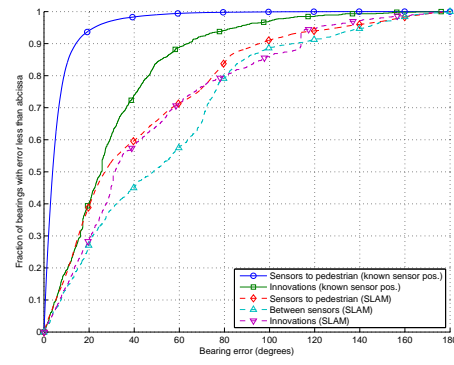


(h) Random 2

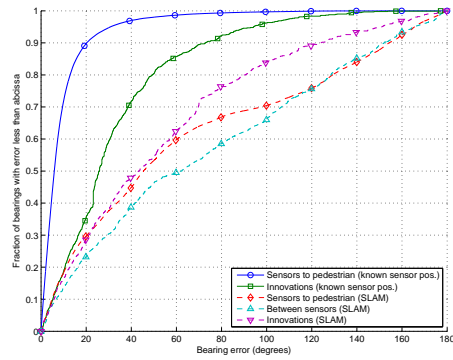
Figure 5.12: Cumulative range error distributions for the full duration of each experiment.



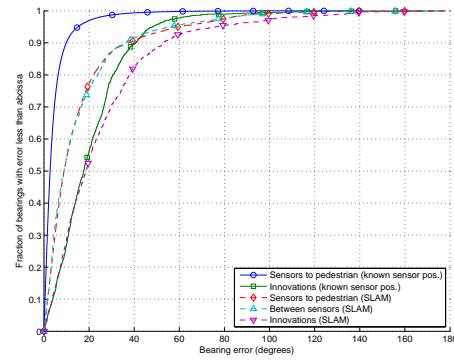
(a) Large room



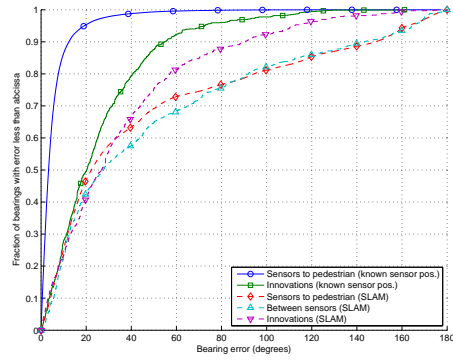
(b) All rooms (T) 1



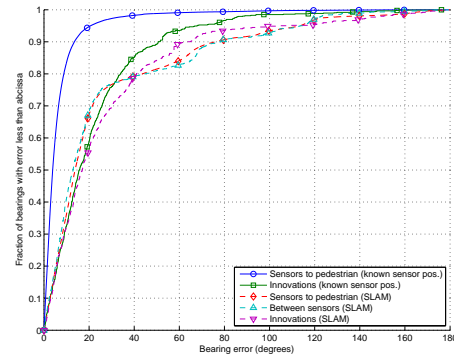
(c) All rooms (T) 2



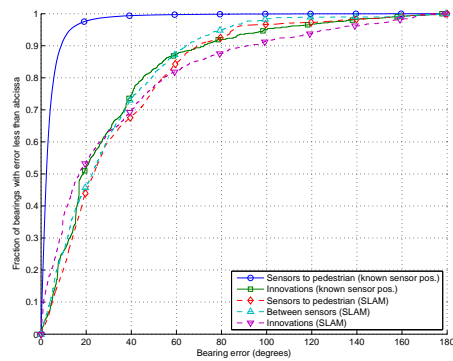
(d) All rooms (T) 3



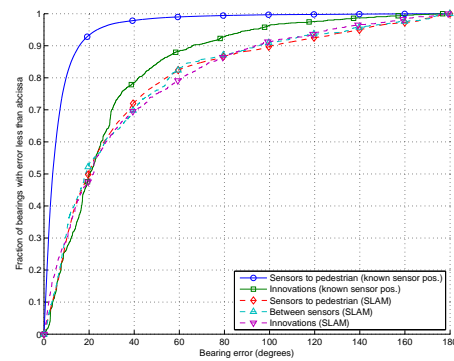
(e) All rooms (T) 4



(f) All rooms (O)



(g) Random 1



(h) Random 2

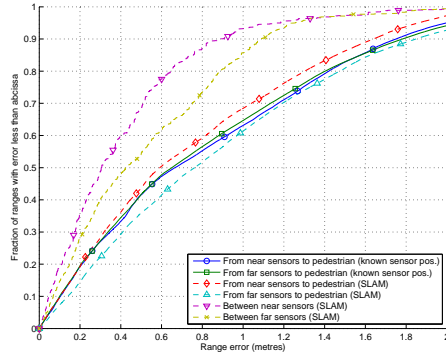
Figure 5.13: Cumulative bearing error distributions for the full duration of each experiment.

simple geometric properties, distance dilutes the effect of position error on the bearing. SLAM bearing errors are generally large. In some cases, despite our heuristic initialisation method (fig. 5.3), the SLAM algorithm places sensors on the wrong side of a straight section of path due to the symmetry ambiguity (experiments (c) and (e) especially). This does not affect local range errors between the sensor and the pedestrian because of the symmetry, but the errors for far away sensors are increased. Sensors in this situation are likely to have very inconsistent orientations and thus the bearing errors for both near and far away sensors are high (figs. 5.15(c) and 5.15(e)). This ambiguity is difficult to resolve with our current implementation because only the sensors take bearing measurements to the pedestrian. If the pedestrian-worn sensor took measurements to the deployed sensors, then these errors could more easily be avoided.

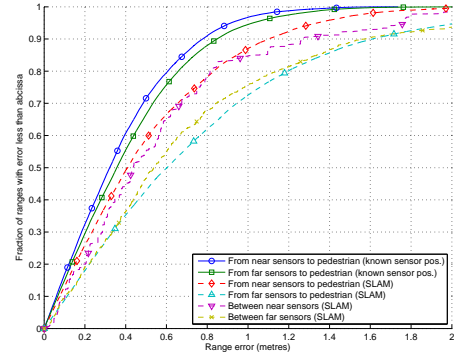
All our experiments can be split into sections of similar duration during which a similar path was walked. For experiments (b) to (f), the same path was repeated several times. For experiments (a), (g) and (h), the pedestrian returned to the same point at regular intervals. Whereas the previous figures show the aggregate errors over the full duration of the experiment, figures 5.16 and 5.17 give the median range and bearing errors for each section of the path. In a few cases there is a noticeable improvement at each iteration but for the other cases the median error remains constant or even increases. For these latter cases, this could mean that sensor and pedestrian position estimates are as good as they are going to get after the first section and that there is no further improvement. After a firefighter has explored the building once and deployed the sensors, the system is immediately ready to help the following teams find their way.

5.3.3 Results summary

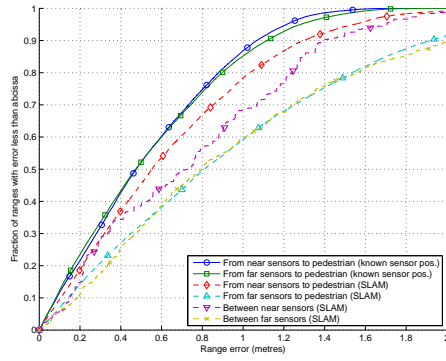
We evaluated our implementation of an ultrasound-assisted pedestrian tracking system by comparing the ranges and bearings computed using the estimated po-



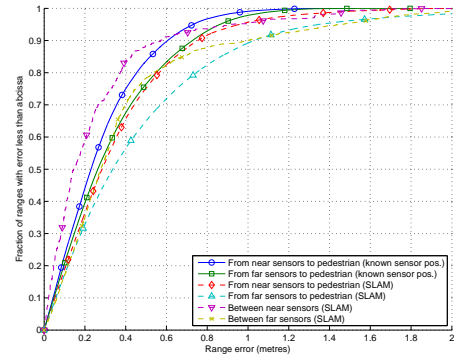
(a) Large room



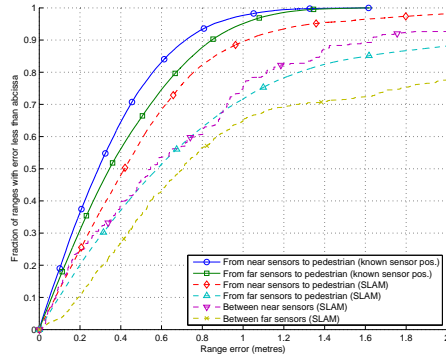
(b) All rooms (T) 1



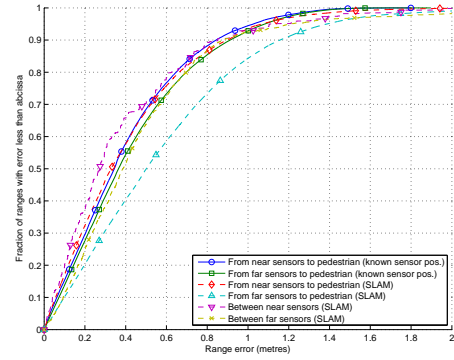
(c) All rooms (T) 2



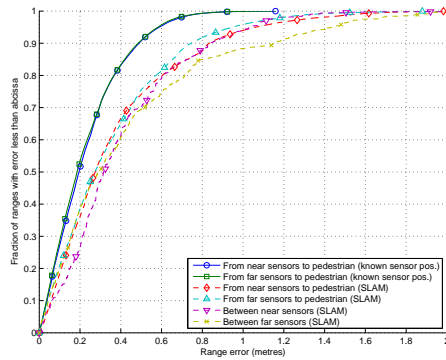
(d) All rooms (T) 3



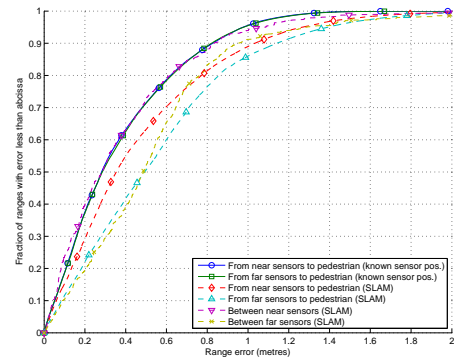
(e) All rooms (T) 4



(f) All rooms (O)

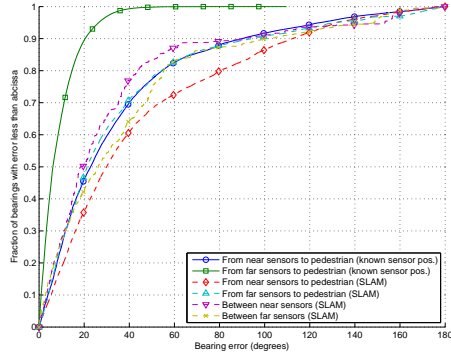


(g) Random 1

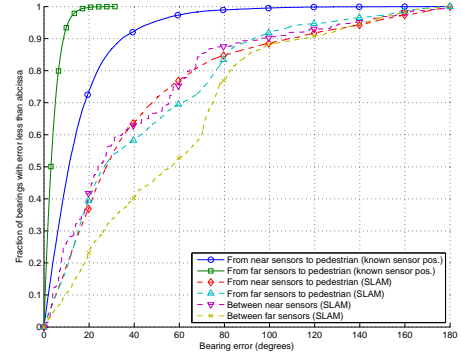


(h) Random 2

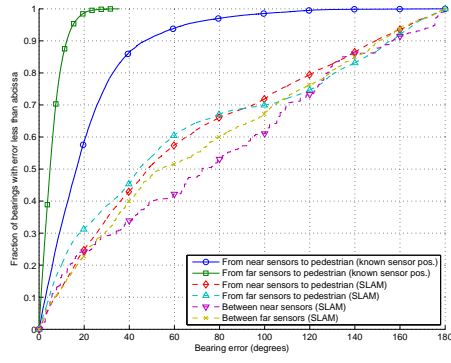
Figure 5.14: Cumulative range error distributions for near sensors ($\leq 3\text{m}$) and far sensors ($> 3\text{m}$).



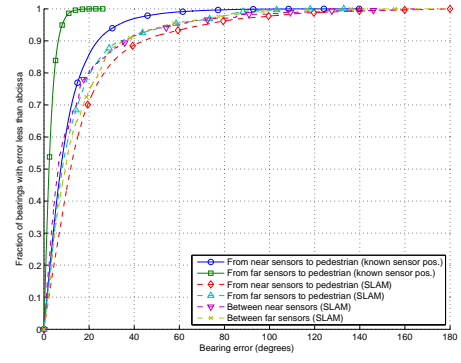
(a) Large room



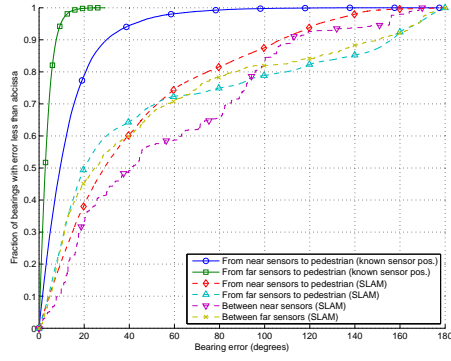
(b) All rooms (T) 1



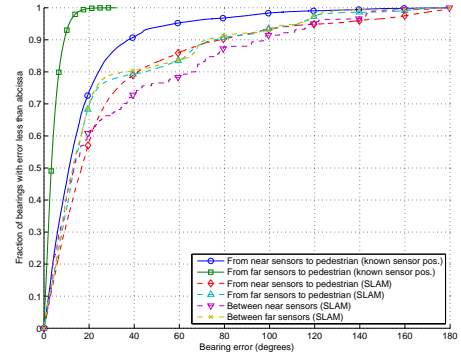
(c) All rooms (T) 2



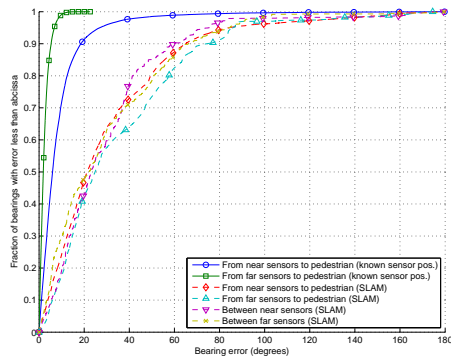
(d) All rooms (T) 3



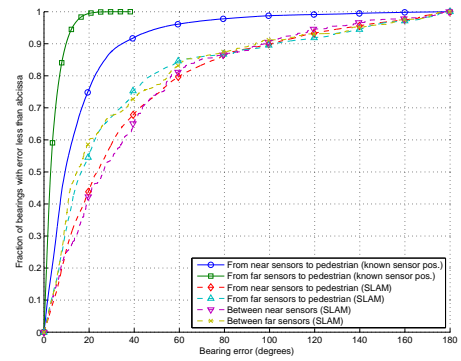
(e) All rooms (T) 4



(f) All rooms (O)



(g) Random 1



(h) Random 2

Figure 5.15: Cumulative bearing error distributions for near sensors ($\leq 3\text{m}$) and far sensors ($> 3\text{m}$).

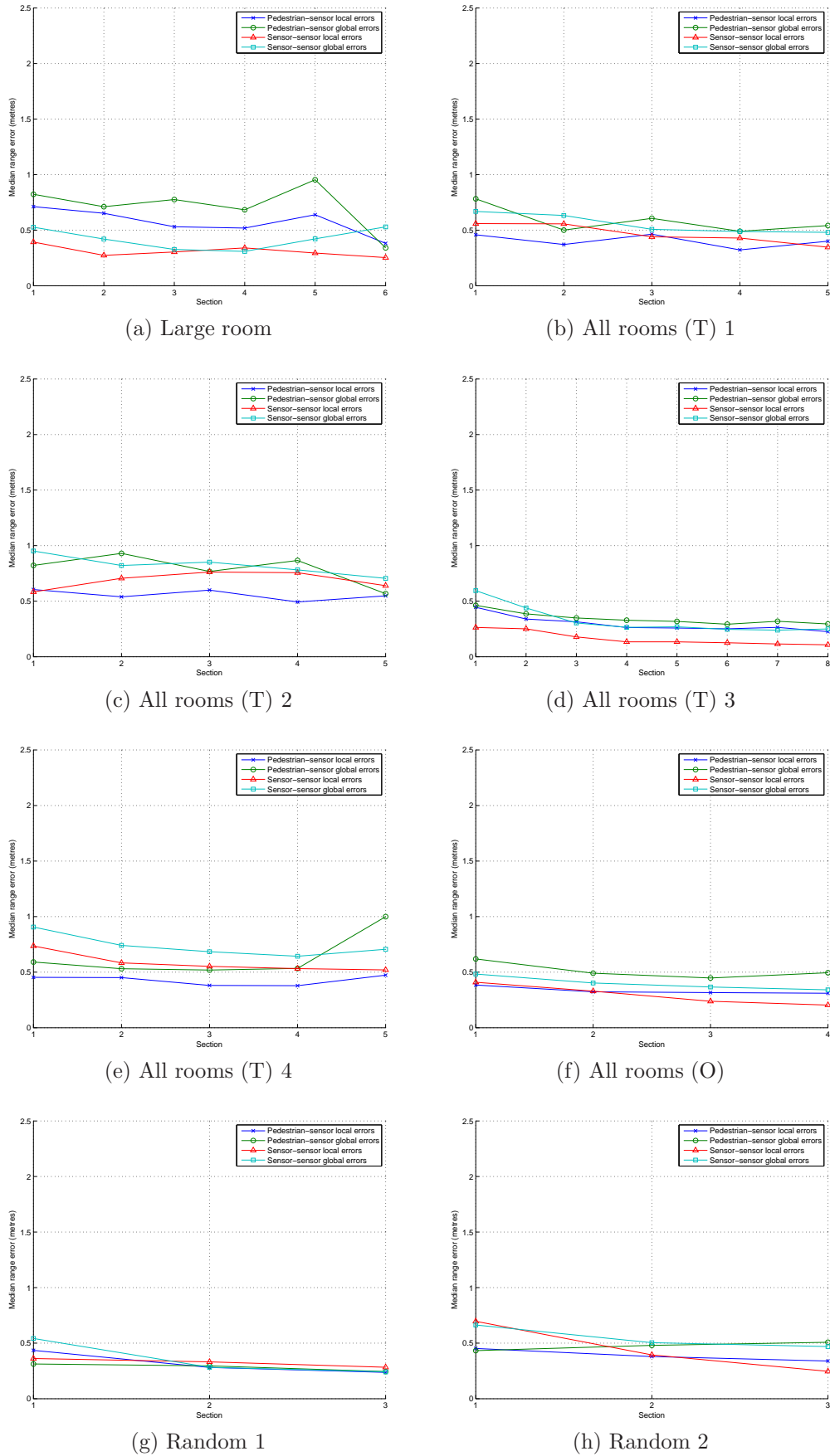
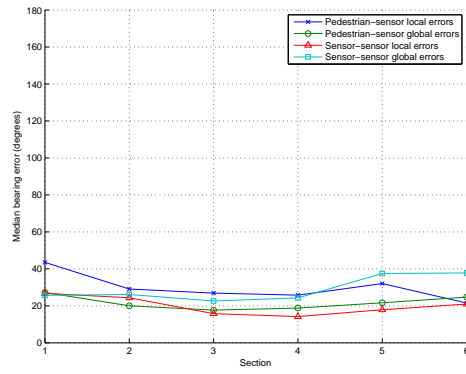
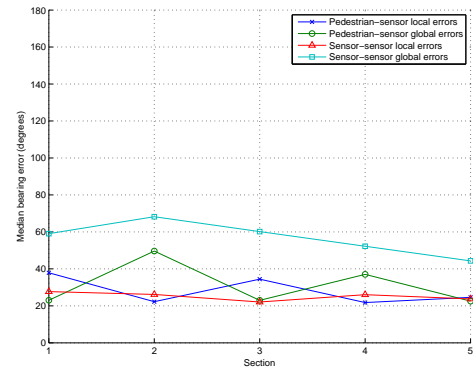


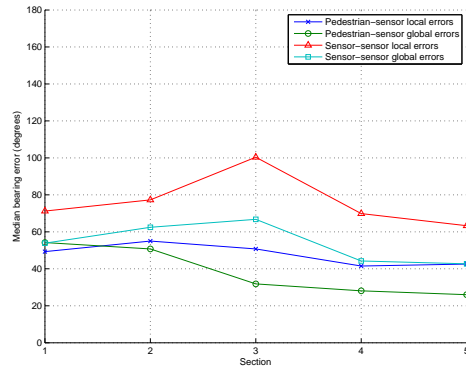
Figure 5.16: Median range errors for each successive section of the path.



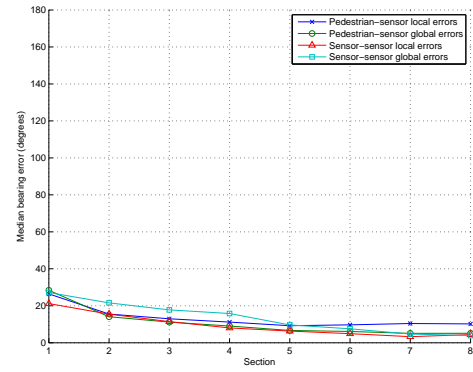
(a) Large room



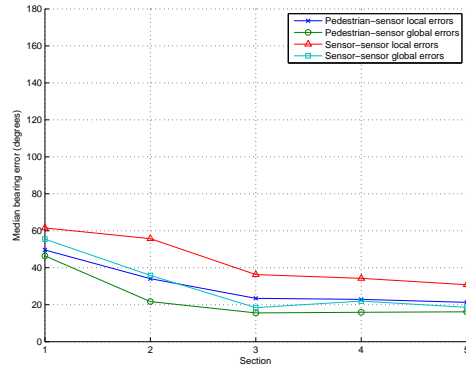
(b) All rooms (T) 1



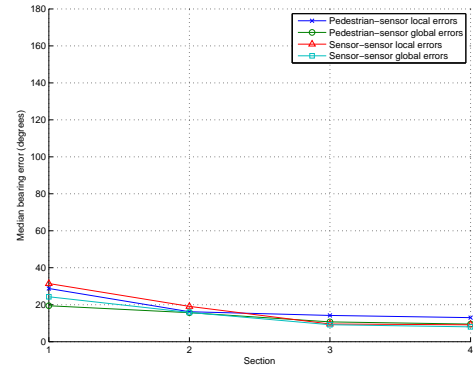
(c) All rooms (T) 2



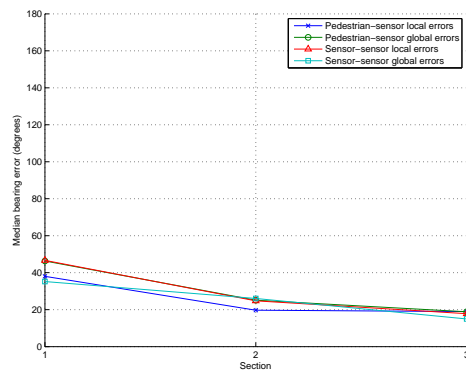
(d) All rooms (T) 3



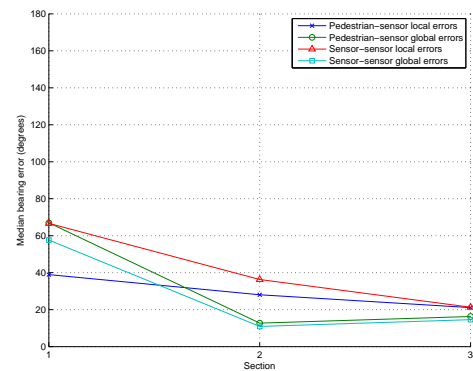
(e) All rooms (T) 4



(f) All rooms (O)



(g) Random 1



(h) Random 2

Figure 5.17: Median bearing errors for each successive section of the path.

sitions of sensors and pedestrian with the ranges and bearings computed using the groundtruth positions. Our implementation achieves a median relative position accuracy of between 0.3 and 0.6 metres when the sensor positions are known in advance. When the sensor positions are unknown and we perform SLAM, our implementation achieves a median relative position accuracy of between 0.4 and 0.8 metres. In the SLAM version, some of this error is caused by large scale distortions in the map (the estimated positions of the sensors). We have shown that the relative position errors between the pedestrian and sensors closer than three metres are comparable to the errors when the sensor positions are known.

Correct initialisation of sensor positions is essential in order for the SLAM method to perform well. It appears to perform better when the paths are unstructured (not following a sequence of straight segments). We believe this is primarily because sensor positions are more likely to be initialised correctly when the path contains many turns than when it is mostly straight. MDS-MAP which we used to initialise sensor positions in chapter 3 is not applicable here because we want to initialise each sensor as soon as possible without waiting to first observe all sensors.

For the SLAM method, there is no clear improvement of sensor position estimates as paths are repeated. However the sensor orientations do improve with time. This result suggests that this type of SLAM system would work well if it was used in emergency response scenarios where the sensors are deployed on-the-fly, and where paths are not necessarily repeated many times.

Finally, our results suggest that some of the error which appears in our plots is due to inaccuracies in the ultrawide band reference localisation system. This could mean that our system actually performs better than indicated by the results presented in this chapter.

5.3.4 Improvements and future work

Although we believe the results we have presented confirm the viability of this type of hybrid tracking and navigation system, there are a number of areas which require further work. Here we discuss a few avenues for further investigation and improvement.

Tracking orientation. At the time of the experiment our PDR system did not keep track of the orientation of the pedestrian, but only their direction of travel. Estimating the orientation of the pedestrian is a requirement if we are to provide them with navigation assistance, since they need to know which direction their target is in. There is a simple relationship between the direction of travel and the orientation of the inertial sensor, but a number of challenges remain. First, the orientation of the foot-mounted sensor does not accurately represent the orientation of the pedestrian or of the graphical interface that we are using to display our map. In order to measure this, we would need to attach additional inertial sensors to the head or chest of the pedestrian. Second, in our current SLAM system, the orientation (and direction of travel) can only be corrected when the pedestrian is moving, by using the cross-correlation between the error in the estimated direction of travel and the error in the estimated position (an error in the estimated direction of travel will cause an error in the estimated position). This could be improved by directly measuring the bearing from the pedestrian to the landmarks (e.g., landmark X is behind the pedestrian). (Currently we only measure bearing from the landmarks to the pedestrian, e.g., the pedestrian is to the right of landmark Y.) Finally, it is challenging to evaluate how well the orientation is being tracked because very few localisation systems which can be used as groundtruth provide reliable orientation (a digital compass may work quite well in open areas but will become unreliable in buildings). Tracking the orientation of the pedestrian remains an essential part of a navigation system, so it is important to improve our system in this respect. Since these experiments were run, we have successfully implemented

an inertial tracking system using one foot-mounted sensor and one head-mounted sensor. We are able to track the position using PDR while also estimating the direction the wearer is looking.

Multiple sensors. Groups of mobile sensors (both inertial and ultrasonic) could be used to great advantage. These sensors could be installed on both feet of the pedestrian or on multiple pedestrians traveling together. We anticipate that such scenarios would not only provide a lot more data than a single sensor in the same amount of time, but would also benefit from the fact that the errors on each sensor are independent from the others. For instance, PDR errors are typically due to thermal noise and small vibrations in the MEMS components which will be different for each sensor particularly if they are mounted on different feet or different people. The SLAM algorithm would also benefit from additional measurements from slightly different positions which could overcome many situations where the line of sight is blocked by the other foot. The challenge would be to initialise all the groups of sensors correctly so they all start moving in the same direction. GPS could be used to synchronise the movements of sensors on different team members before they enter a building. If GPS is not a suitable solution, maybe range and bearing measurements between pedestrians could be used to locate them relative to one another. Strömbäck et al. (2009) have started investigating such a cooperative navigation system using ultrawide band radios for ranging between two pedestrians.

Loop closure. We have shown that some of the error in the SLAM output was due to large-scale distortions in the map. These distortions can often be corrected when the path contains large loops. Gutmann and Konolige (1999) and others have shown that loops can be used to correct maps built from dense laser range scans. Many of the challenges they describe are non-existent in our implementation because we use sparse landmarks with explicit identifiers. Unfortunately, our data does not contain such loops because each room only had a single entrance,

so we have not been able to validate this aspect of our system. In principle, loops are not a problem for our SLAM algorithm but the map will not necessarily be correct until the loop has been walked several times, and, depending on the magnitude of the error, there may be stability issues. Graph-based optimisation (Thrun and Montemerlo, 2006; Kleiner and Sun, 2007) or backward correction of path estimates (Thrun, 2001) offer solutions to the loop closure problem.

Alternative Bayesian filter implementations. We have noted problems due to the non-linearities in the process and measurement functions which cause the location estimate error to be underestimated. These could possibly be solved by using alternatives such as a particle filter or an unscented Kalman filter. The particle filter would also have the benefit of resolving the ambiguity in the initialisation of sensor positions since it inherently allows multimodal solutions. The initialisation problem may also be solved more simply by having the pedestrian deploy the sensors just next to their feet so that the sensor positions can be directly initialised using the estimated position of the pedestrian (see chap. 6).

Evaluation methods. The evaluation of such infrastructure-less tracking and navigation systems is problematic. First, it is difficult to obtain any accurate ground truth for large areas inside buildings. Second, even when accurate, high-update rate, wide-area ground truth is available, it cannot easily be aligned to the estimates due to coordinate systems which change over time. Some of the methods we have used in this chapter are useful evaluation tools, but we hope that future research will provide additional ways of comparing different navigation systems before passing them over to users for testing, both in the form of technologies for recording ground truth positions over wide areas and experimental best practices. In the following chapter, we investigate virtual reality simulations as one possible way to achieve this.

Practical considerations. Finally, the conditions in which such systems will be used will require very careful design of both the hardware and the procedures

for its use. Inertial sensors are sensitive to variations in temperature but we do not know whether the effects will impair the performance of the system, or whether they can be successfully mitigated by thermal stabilisation measures (e.g., filling the sensor enclosure with oil) or software compensation. The environmental conditions will also affect the propagation of ultrasound but the exact consequences have not yet been determined experimentally. In the work presented in this chapter, we also did not examine how the beacons would be deployed in practice. It could make the initialisation phase of the algorithm more reliable if we are able to approximately initialise the landmark positions with the position of the user at the instant they were deployed. This depends on exactly how the beacons are deployed, whether they are automatically ejected or placed by hand, how far they roll or bounce, the density of the network and many other practical issues.

5.4 Conclusion

In this chapter, we focused on the use of a sensor network to provide *positioning* and *tracking* capabilities that could potentially support emergency responders. Although the application of sensor networks to support emergency response, and in particular firefighting, has been explored in a range of projects by pre-deploying positioning infrastructure, our research contrasts by using an ad-hoc approach that does not require any pre-deployment.

Based on the understanding of the errors encountered in the PDR location estimates, we looked into complementary technologies that can correct the drift. Specifically, we used ultrasound sensors which have the capability to measure relative range and bearing between the tracked pedestrian and the deployed sensor nodes. We used an extended Kalman filter (EKF) *simultaneous localisation and mapping* (SLAM) method to concurrently estimate the location of the deployed sensor nodes and the position of the pedestrian. Fusing ultrasound measurements

with inertial tracking enables us to prevent the drift which makes inertial tracking unreliable when used alone. In all but one of our experiments, ninety percent of range errors were less than two metres, even when no prior information about the environment was available.

The deployment of small sensors during an intervention is consistent with practices in certain firefighting units where a first team searches a building before further teams enter to assist victims and attack the fire. Although there is plenty of room for improvement, we believe our EKF implementation provides location estimates suitable for indoor navigation of emergency responders and shows that the combination of ultrasound sensors deployed on-the-fly with inertial pedestrian dead reckoning is a viable solution. We strongly believe that such a combination of modalities (inertial and ultrasound) can be extended to provide a fully functional ad-hoc positioning system for guiding emergency responders. Although these technologies and algorithm can be used to provide navigation support to team members inside a building, they are not immediately suitable for providing an overview to a third party such as the incident commander located outside. This would require further investigation into ways of presenting the data on top of floorplans or satellite imagery for instance, and merging maps from multiple teams.

This chapter focused on the design of the system, and we used real world data to show that this could be a practical navigation solution. However, there remain many avenues for improvement in terms of sensing hardware, SLAM algorithms, and high level system design. In chapter 6, we use this particular SLAM concept as a basis for navigation support for firefighters in a high-fidelity simulated environment.

Chapter 6

Design and evaluation of a navigation system using a virtual reality simulator

The previous chapters have given us the building blocks for a navigation system suitable for use in an unprepared environment for which we have no maps and in which GPS is unavailable. Experimental results are promising but the practical implementation of such a navigation system will present issues beyond the sensors and the algorithms. The way in which the sensors are deployed, how the navigation information is presented to the user, how they respond to it, and how their behaviour affects the system, are all areas which are missing from most research into localisation. We investigate those aspects in this chapter.

A fundamental question is how we can evaluate a navigation system without building a complete and realistic environment to test it in. We distinguish *navigation* systems from *localisation* or *tracking* systems which do not present such a challenge. The position estimates from a tracking system can easily be compared to the ground truth positions in an absolute coordinate system, and the errors used as a metric to evaluate the performance of the system. Measuring the ground truth can be a challenge but it is merely a technical one. Navigation, however, presents a more fundamental challenge because there is a human *in the loop*. As

human beings, we all have a certain sense of direction, some more than others, and we are able to find our way in many different environments, known and unknown. Sometimes, our own navigation strategies and common sense are sufficient to make up for substantial errors or missing information from a navigation system. Other times, small errors or omissions in the system can confuse us and lead to our getting lost; for instance, depending on the situation and the person, an incomplete map could make finding their way easier or more confusing. This is difficult to predict without testing a full implementation of the navigation system in question, and the answer is heavily dependent on the way the information is presented to the users.

In this chapter, we show how we can answer some of these questions and accelerate the development of real solutions to our navigation problem through virtual reality simulations. These video game-like simulations allow us to try out a variety of ideas for navigation systems and test them with real users without time consuming and expensive hardware development. This method is no substitute for full system testing towards the end of the development process, but we believe it helps fix a lot of the unknowns in this type of system development, and reduces the number of design iterations that need to be implemented *in the real world*.

6.1 Introduction

We use a virtual reality simulation, essentially a video game, to evaluate some of the ideas presented in the previous chapters. The simulation allows us to gain more insight into how viable these ideas are, in particular with respect to a user's behaviour when faced with an imperfect system. This is a useful tool for researchers who do not have the resources to develop new hardware or the time to run many full-scale user studies.

6.1.1 Evaluating navigation systems

Several researchers have successfully evaluated novel navigation interfaces by running user trials in the real world, for instance Rukzio et al. (2009), and Heuten et al. (2008). The first study takes place in full visibility and the participants follow directions along a path. The second study is more relevant for us because the participants are blind-folded. However, it takes place in an open field where there is little risk of injury. Walker and Lindsay (2006) test an auditory navigation system in an immersive virtual environment and write that participants who used both the physical prototype of the navigation system and the virtual system did not report any major differences between the experiences. Witmer et al. (2002) highlight the potential of virtual environments for training users in dangerous environments, but their studies also use a fully immersive system with a head-mounted display and additional tracking of the players' walking movements. It is therefore not immediately clear that our desktop-based simulation presents the same benefits. We have not found conclusive evidence in the literature that evaluating a technology in a virtual environment is a valid substitute for real-world evaluation, especially when the virtual environment is non-immersive (i.e., a basic flat screen as opposed to a head-mounted display or a panoramic screen). Nonetheless, at the very least, we have found that this study has been a worthwhile exercise which has forced us to think about certain aspects of the implementation, and which has given us fresh ideas for corrections and improvements.

Our work in this chapter offers the following benefits:

- It allows the (virtual) use of technology which is unavailable due to lack of resources or limitations in state-of-the-art hardware.
- It avoids the risk of users tripping and injuring themselves during a study conducted in low-visibility conditions.

- It permits testing of a range of navigation technologies and methods such as sensor nodes, pedestrian dead reckoning and machine vision.
- It encourages concrete advances in sensor hardware by providing specific application requirements.
- It allows us to evaluate navigation systems while taking the user and their innate navigation skills fully into consideration.

The virtual reality simulations we propose have their inherent limitations too. Implementing various navigation systems in the virtual environment requires some effort and creativity to overcome implementation issues. However careful and precise the simulation, the real sensors will behave differently from their virtual counterparts, and a person will behave differently in the real world from when they are playing a video game. One challenge, common to all user studies and not addressed by our suggested evaluation method, is the large number of users required to compare the effects of different parameters. There is also a strong learning effect if users navigate the same environment several times, so we either need more users or many different environments. Despite these issues, we believe this type of simulation to be a helpful tool in discovering practical implementation and usability issues.

6.1.2 FireSim

We base our work on FireSim, a virtual environment designed by researchers from Fraunhofer FIT to better understand the work practices of firefighters. Klann (2007) describes how he and his colleagues used first low fidelity board games and then the virtual reality simulator, FireSim, to design a wearable computing solution for firefighters. Their current work includes mixed reality simulations where FireSim is connected to real radios, head-mounted displays and wearable computers, as they

become available. Klann and his team ask the firefighters to play out specially designed scenarios in order to determine which aspects of the support system are successful and which need further work.

Our work differs from the work of Klann by its focus. While he is concerned with the overall system and how it fits into the existing work processes, hierarchies, and expectations of the firefighters, we see FireSim as a tool we can use to investigate specific localisation and navigation methods, independently of the higher level constraints. The emergency response and search-and-rescue application has helped us define a number of criteria that are important for the design of our system, but once these have been established, we prefer to work at a more technical level before taking into consideration the higher level requirements which are beyond the scope of this thesis.

6.1.3 Contributions

Our contributions in this chapter are as follows:

- We demonstrate a novel navigation system that does not depend on existing infrastructure. This builds on the tracking system from the previous chapter.
- We show that a virtual reality simulation is a useful tool for testing and evaluating novel methods and algorithms for localisation and navigation.
- We use our simulator to evaluate a simultaneous localisation and mapping (SLAM) system of our design based on ultrasound beacons and pedestrian dead reckoning (PDR).
- We provide insight into how people use the navigation support we provide and how the system could be improved.

6.2 Navigation system evaluation

We conducted a user study to evaluate the effectiveness of our novel navigation system, both the interface and the underlying algorithms. Participants were asked to play a video game in which they took on the role of a firefighter performing a search and rescue mission in a dark building. This also provided the opportunity to observe their behaviour when faced with a study based on a video game.

6.2.1 Study overview

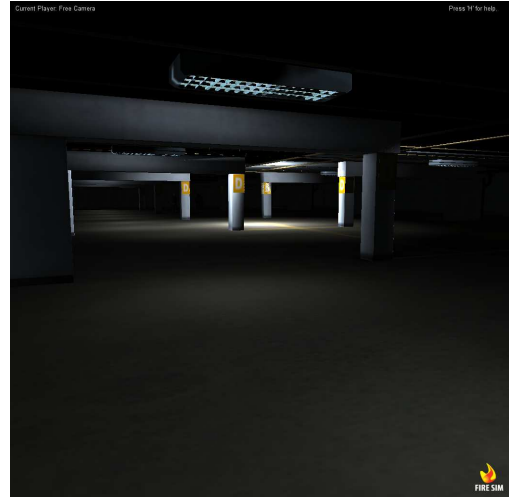
During our user study, we asked each participant to accomplish a simple mission in the virtual environment. They were asked to search a building for a missing person. When they found the person, they were told to return to their starting point. They each performed this mission three times in an underground car park, and three times in an office building (fig. 6.1). The car park is essentially an open space, while the office building is much more structured with corridors, and offices of different sizes. In each building, one mission is performed with no navigation support, one mission with the *arrow* interface, and one mission with the *map* interface. These interfaces are described in more detail in the following subsection. The order in which the two buildings are presented alternates between users, and the order of the interfaces is balanced (see appendix C). If the player does not find the missing person within three minutes, they are given the signal to retreat to their starting point.

To ensure that all players search for a reasonable amount of time, we only place the missing person in the game after 90 seconds in a location that the player has not yet searched, but the player is not made aware of this. They are given a total of 180 seconds to complete the search. These timings are designed so that a very efficient player will be able to find the missing person while keeping the total study length

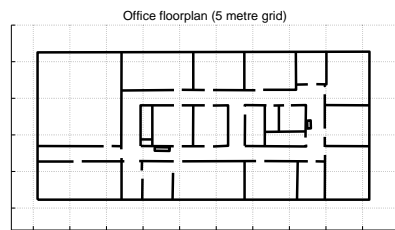
reasonably short. A good search strategy still pays off because it will constrain the missing person to appear in a more limited set of locations. Once they have found the person, or if they run out of time, the missing person is removed from the game to avoid any distraction during the retreat phase. There is no time limit while the player searches for their starting point.



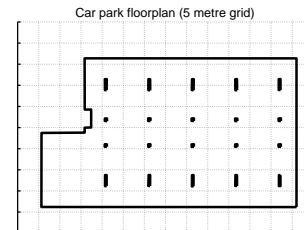
(a) Structured office-type environment with corridors and small rooms.



(b) Unstructured underground car park.



(c) Office building floorplan.



(d) Car park floorplan.

Figure 6.1: Structured and unstructured environments in the virtual reality simulation. Floorplans as presented to the participants.

The starting point is different for each mission but is always selected from a predefined list of suitable points. It is always close to an external wall and is marked with a traffic cone to enable to player to identify it unambiguously. The movement of the character in the game is between five and six kilometres per hour. This is close to, or slightly higher than, the average walking speed for most people, but is much slower than the speeds at which characters move in many first person video games. The player can move forward and backward, and turn, but not sidestep. The limited speed and controls are put in place to prompt a more thoughtful

and realistic strategy from the players. In order to prevent players from walking in perfectly straight lines using the keyboard, we add some offset to the controls which causes them to deviate slightly.

Our participants were recruited from among the university students and staff, excluding anyone who was already aware of our research or conducting similar research. They were paid £8 for their time. Half the participants were male, half were female. The ages ranged from 21 to 30. Two participants suffered from motion sickness during the study. One retired from the study towards the start so their data was discarded and they were replaced by another participant. The other only needed to leave briefly at the very end of the study but returned to complete all questionnaires. The total duration of the study for each participant was less than one hour.

6.2.2 Navigation system

The navigation system included in the game is based on the underlying idea of this thesis: that we need to provide navigation support that is not dependent on any existing infrastructure or prior knowledge about the building. The previous chapter explores a particular implementation of simultaneous localisation and mapping (SLAM) which allows us to build a map of sensor nodes with initially unknown positions while simultaneously locating a pedestrian walking among them. In this chapter, we extend this scenario and make it more relevant to real applications by deploying the sensor nodes as the pedestrian progresses and feeding the location estimates back to them to assist them in their navigation. We use the same Kalman filter-based SLAM algorithm as described in the previous chapter to estimate the positions of the pedestrian and of the sensor nodes, also referred to as beacons due to their high visibility. We simulate the range measurements from the beacons to include some additive Gaussian noise, and the inertial movement measurements include a non-zero mean Gaussian error which causes the orientation and direction

of travel estimates to drift, as is the case with actual PDR position estimates. A new beacon is automatically deployed when necessary, i.e., when the pedestrian has moved far enough from the other beacons. In addition to the positions, we build a tree (graph) which connects all the beacons. The root of the tree is the starting point, and each beacon knows which other beacon is one step closer to the root of the tree. The pedestrian is always virtually “connected” to the *nearest beacon*. The algorithms for building the tree and keeping track of the relevant beacons are given in appendix B.

The *arrow* interface is designed to be very simple (fig. 6.3c). The tree structure is hidden from the player. They only see a green arrow which points to the next beacon which will bring them closer to the starting point. This beacon is referred to as the *target beacon*. The green arrow effectively guides the player back along their path. A red arrow points directly to the starting point, regardless of the path previously taken by the player; it is designed to help them remember the general direction of the starting point and take a short cut if they wish. This was requested following the pre-study. As the player moves around, the navigation system constantly recomputes the position of the target beacon relative to the pedestrian and orients the green arrow accordingly; it also tests whether the player has reached the target beacon and should now be guided to a new target. Both the arrows are continuously updated to be consistent with the player’s orientation in the game, with the typical convention in which *up* represents the *forward* direction. Figure 6.2 illustrates how the player is guided and details of the algorithm used are given in appendix B.

Our *map* interface displays the tree structure graphically (fig. 6.3d). Each node is shown at its estimated coordinates, and connected to its parent and child nodes by means of lines with arrows. The pedestrian is shown in the centre of the display and connected to the nearest beacon via a similar line. The arrows on the lines always direct the player back to the starting point which appears as a star on the

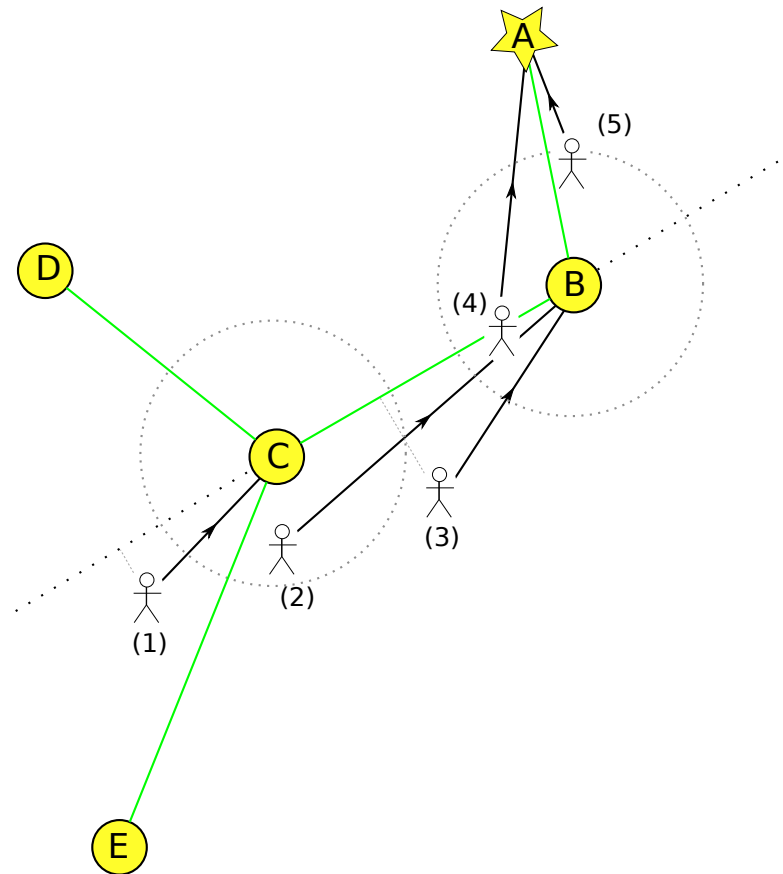


Figure 6.2: The player is guided along a path of beacons to the exit (A). The arrows indicate which beacon the pedestrian is guided to at a given position. If they are within a certain distance from the nearest beacon, they are guided to the next beacon (2,4,5). If not, we test whether they are between them (3) or not (1). (See appendix B for full algorithm.)

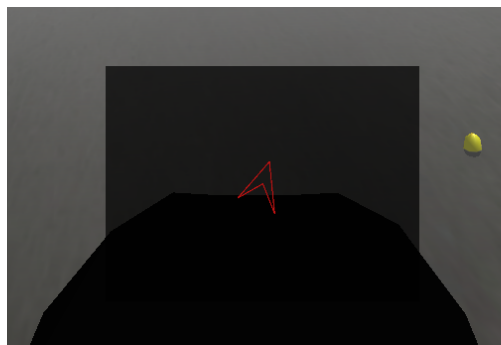
map. If the player backtracks before heading off in a different direction, a new branch will be created on the map (and in the underlying tree data structure). On the map, the player's position is surrounded by a circle of radius one metre corresponding approximately to the area within arm's reach, for scale. The orientation of the map is updated in real time according to the player's orientation.

Neither of these interfaces require or suggest any prior knowledge of the environment. In fact, at the start of each mission, the map is empty with only the starting point represented by a star in the centre of a blank area. The map builds up as the player walks around. Although it does not tell them how to search the building, it shows them where they have been and where they have not yet explored. Its main use comes when the player needs to return to the starting point: they simply follow one of the arrows or the map lines.

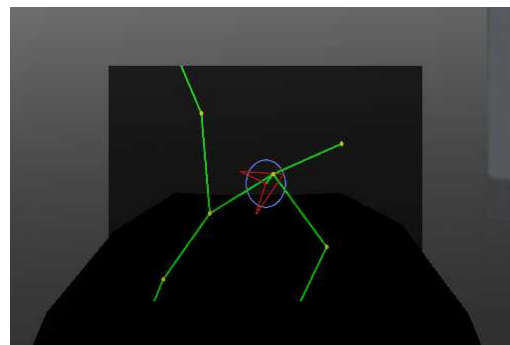
We deliberately limited the area visible on the map to a relatively small area around the player (approximately 12×20 metres). We saw in the previous chapter that the position estimates can be good locally, but poor on a larger scale. From the player's point of view, this means that even though the complete map might appear incorrect (e.g., straight lines appearing as curved or vice versa), the close up view of any portion of the map may still appear as expected (e.g., a left turn at the expected place).

6.2.3 Study procedure

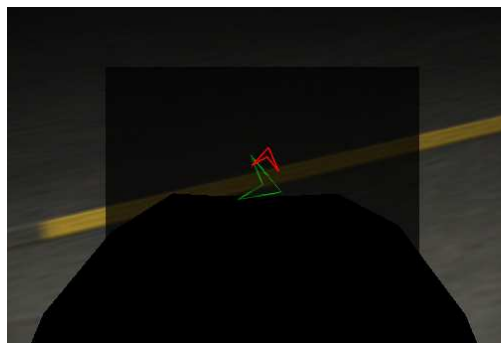
Our users play the game individually, one at a time, on a laptop computer, using either a keyboard and mouse, or an XBox 360 gamepad, according to their preference. After explaining that we are investigating navigation in low visibility, we describe the search task. We emphasise that the navigation system is not perfect, and that the user should feel free to follow their own instinct if they feel the system is incorrect or if they do not need it. Then we launch a training session of



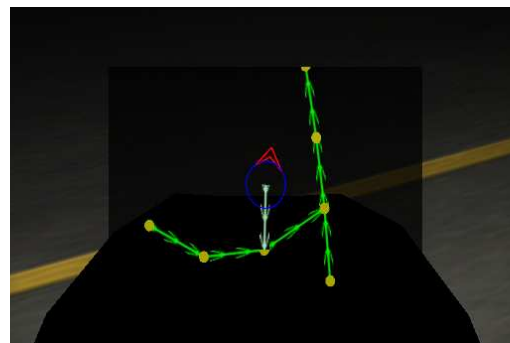
(a) Arrow navigation interface in pre-study.



(b) Map navigation interface in pre-study.



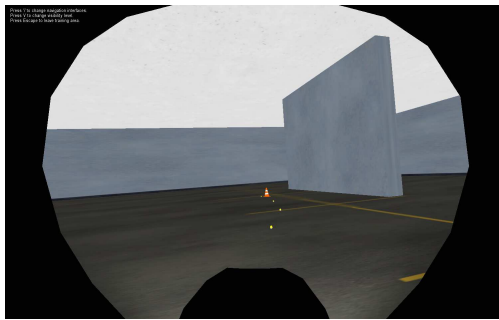
(c) Arrow navigation interface in main study.



(d) Map navigation interface in main study.

Figure 6.3: Close-up showing changes to the navigation interface following feedback from the pre-study. We added direction to the map lines and a direct path arrow, changed the colour of the smoke, and added a head torch.

the game where we demonstrate the controls and show how the navigation system works. We give the player a few minutes to walk around in the training area so they can familiarise themselves with the controls and the navigation system. During the training session, we decrease the visibility until it is the same as in the game itself (fig. 6.4).



(a) Training area with good visibility (navigation interface disabled).



(b) Training area with poor visibility (navigation interface disabled).

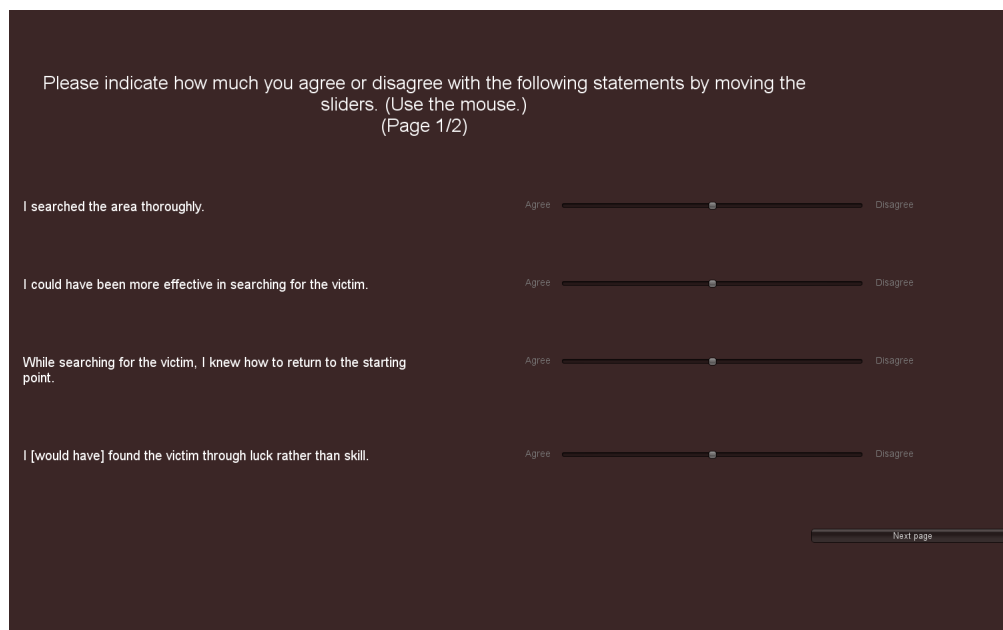
Figure 6.4: The training area allows the players to familiarise themselves with the controls and navigation system before starting the study proper.

Before the player starts the first mission in each building, we show them a floorplan of the building and give them approximately 20 seconds to look at it while we describe what the building is like (figs. 6.1c and 6.1d). We highlight the fact that the carpark is not a complete rectangle, and that some offices have two doorways. The floorplan is not available to them at any other point of the study. This is intended to even out the learning effect by familiarising the player with the layout of the building in advance, thus reducing poor performance during the first mission. We adopt this approach because Witmer et al. (2002) record several navigation studies in virtual environments where the players exposed to maps prior to the study perform at least as well as those who are not. We also explain that many firefighters are trained to keep one hand on a wall and to follow this wall to avoid getting lost. Again, this is intended to provide all players with at least one reasonable strategy, thus evening out some of the differences between players.

When the mission starts, the player is at their starting point, next to a traffic cone which serves as a marker. They explore the area as they choose. When they

find the missing person, or when they run out of time, the word ‘retreat’ flashes on the screen several times and they return to their starting point as quickly as possible. When they have found it, they are presented with a questionnaire about the mission. They respond on the screen by moving sliders to indicate whether they agree or disagree with the each of following seven statements (fig. 6.5):

1. “I searched the area thoroughly.”
2. “I could have been more effective in searching for the victim.”
3. “While searching for the victim, I knew how to return to the starting point.”
4. “I [would have] found the victim through luck rather than skill.”
5. “While retreating to the starting point, I felt disoriented.”
6. “I reached the exit in the most straightforward way.”
7. “I was confident I would find the exit.”



The screenshot shows a questionnaire interface with a dark background. At the top, it says: "Please indicate how much you agree or disagree with the following statements by moving the sliders. (Use the mouse.) (Page 1/2)". Below this, there are four statements, each followed by a slider control. The statements are: "I searched the area thoroughly.", "I could have been more effective in searching for the victim.", "While searching for the victim, I knew how to return to the starting point.", and "I [would have] found the victim through luck rather than skill.". Each slider has "Agree" on the left and "Disagree" on the right. A "Next page" button is located at the bottom right.

Figure 6.5: Questionnaire presented to the player after completing a search mission. They use the sliders to indicate whether they agree or disagree with each statement.

After completing the questionnaire, they move on to the next mission. If the questionnaire is the last one for a particular building, they are also shown a screen

which invites them to discuss their experience with the researcher. These interviews were audio recorded. Overviews of the questions asked and the feedback extracted from the interviews are included in appendix C.

Following the experiment, the path of the player during each mission was overlaid on a floorplan, the search path in blue and the retreat path in red. These traces were anonymised and presented to four researchers familiar with indoor navigation (including ourselves), in different orders, via a website (fig. 6.6). Each researcher rated each trace according to four criteria using a slider:

- Poor search method. – Effective search method.
- Small area searched. – Most of the area searched.
- Lost. – In control.
- Long path back. – Direct path back.

We added the scores given independently by each researcher to determine how well players had performed. The primary benefit of this process is to determine in which cases players performed very poorly by clearly getting lost or walking in circles, for instance. In the pre-study, we used the A* algorithm to find a reasonable path from the retreat position to the starting point while avoiding walls (Hart et al., 1968). We then compared the length of this path to the length of the path taken by the player. The method worked well but we abandoned it in favour of a manual evaluation, because the calculated results do not take into consideration the proximity to the starting point or the fact that the player may legitimately prefer to retrace their steps instead of taking a more direct route, even though the path would be longer.

6.2.4 Pre-study

We ran a pre-study in order to determine whether the players would be able to understand our navigation interfaces and perform the search mission correctly. We

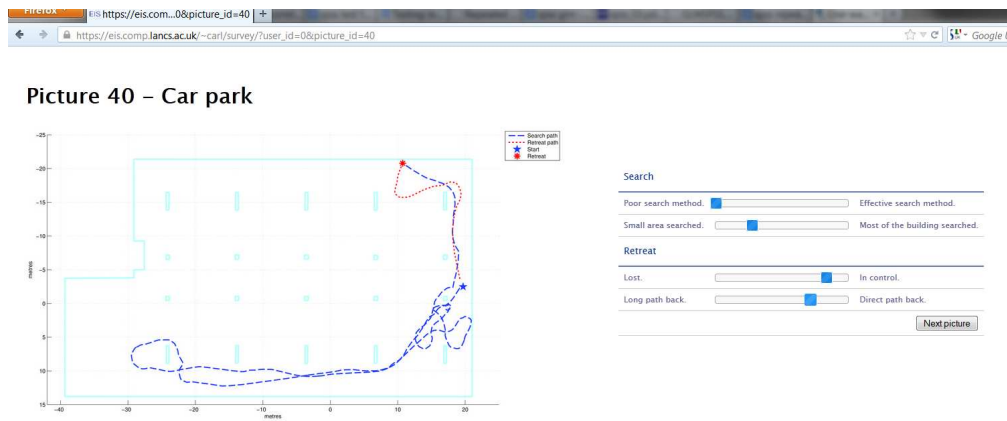


Figure 6.6: Website used to evaluate the players' search and retreat strategies. Traces are anonymised and presented to each researcher in a different order.

recruited six of our colleagues who were aware of our research but had not been closely involved with the development of the study. These participants were not paid. The key difference from the main study was that the navigation support (map and arrow interfaces) was computed using the true positions of the player and beacons in the game. These were obtained directly from the game engine itself, not estimated using noisy measurements as was the case during the main study. This meant the navigation interface was always correct and perfectly stable, allowing us to evaluate the interface itself rather than the underlying SLAM algorithm and sensor data.

Following the pre-study and based on the participants' suggestions, we made several changes to the navigation interface. We added arrows to each of the lines on the map to indicate the direction of the exit, we included a second arrow on both the map and arrow interfaces to indicate the direct path to the exit, and we added a control to manually drop a beacon. This second arrow allowed players to take a short cut to the exit, and we anticipated it would be especially useful in the car park scenario (open space). The option to manually drop a beacon was requested as a way to manually mark a room as visited. The rooms in the office building are small enough that a beacon is not always dropped inside the room if there is already one in the corridor, and, later in the game, players were not sure whether they had searched those rooms or not. The navigation interfaces used in

the pre-study and in the main study are shown in figure 6.3.

6.3 Results

We analysed the answers to the in-game questionnaire and the ratings of the traces to determine the effect of the environment and the navigation interface (presence or absence of navigation support, and the type of navigation interface) on the players. The pre-study gave us some statistically significant results but these could not be reproduced in the main study. We do, however, have some interesting qualitative observations from the interviews.

6.3.1 Pre-study

Preliminary analysis of the pre-study questionnaire data suggested that the participants felt more confident when supported by the arrow or map interface. This was consistent with the verbal feedback from the players. The Friedman test revealed a significant effect of the navigation interface (none, arrow, or map) on the responses to two of the questions for the car park environment: “While retreating to the starting point I felt disoriented” ($\chi^2(2) = 10.174, p = .006$) and “I found the exit due to my resourcefulness (or the instructions) rather than by chance” ($\chi^2(2) = 6.870, p = .032$). However, the post-hoc Wilcoxon test with Bonferroni correction did not detect a significant difference, possibly due to the small number of participants in the pre-study.

6.3.2 Main study

We did not find any visible effect of the navigation interface or the environment on the answers to the questionnaire or on the quality of the search or retreat patterns.

We discuss reasons for this in the following section. However, here we report on the qualitative results for each of the aspects of the study. These are taken from a semi-structured interview performed half way through the study and at the end, after they had completed three missions in each environment. The interviews were audio recorded and annotated later.

Effect of the environment

Ten of the twelve participants said that the office building was easier to search than the car park, primarily due to the structure provided by the rooms. Two participants mentioned that there appeared to be more light in the offices than the car park¹. Another highlighted that it was difficult to know exactly how far they were able to see in the car park by torchlight due to the lack of walls. Nonetheless, one participant described a sense of panic in the office building, while another said that it was more difficult than she expected.

Beacons and other landmarks

Seven of the twelve participants mentioned using existing landmarks in the environment (e.g., fire extinguisher, wooden crate in corridor, sign on wall) to help them navigate. Five participants followed the walls in the car park, and two used the pillars to help them explore the central area away from the walls. Two participants mentioned using the letter 'D' signs on the car park pillars as landmarks, but two other participants told us that, although they had seen these signs, they decided not to use them because they had correctly noticed that they were identical throughout the building²

¹The lighting settings were identical in both environments (no ambient light, only a headtorch worn by the character) but the reflection off the office walls made that environment appear brighter.

²'D' is the level of the car park, not a zone within the level.

Nine of the twelve participants told us that they used the beacons to help them find their way in at least one of the buildings. Five used them to remember where they had already searched. Four participants claimed that the beacons were not helpful or were difficult to use in one environment, but they all said that they had used them in the other. One of the reasons given for them not being useful is the fact that after a while there are beacons everywhere which either create a loop or no longer define a clear path which can be followed back to the starting point.

Arrows

Ten out of the twelve participants told us they used the arrow navigation interface, and two of them said it helped them during the search phase. Three preferred the arrow interface over the map interface because it was simpler to use. These three people used the red arrow which pointed directly to the exit rather than the green arrow which guided them back along their path. However, seven people found the arrow interface confusing in one of the environments, either due to not understanding where it was leading them, or because its direction changed unpredictably.

Map

Ten of the twelve participants told us they used the map interface. Four of these used it during the search phase to remember which areas they had already searched, and two claimed they preferred the map over the arrow interface. However, two participants found the map interface confusing. One of them explained that she could not read a map, the other was confused by lines crossing on the map, either due to estimation errors or a misunderstanding of the interface.

Sample traces

We recorded the paths of the players and overlaid them on the floorplans of the buildings. Note that this does not show the map as estimated by the SLAM algorithm, but it shows the actual path followed by the player in the game. We also recorded the estimated positions of the beacons and the pedestrian, but these are difficult to represent graphically without replaying the whole mission. We can only hypothesise about why the players behaved the way they did in particular situations based on observation of the traces and their feedback during the interviews. Figures 6.7 and 6.8 show a selection of plots which illustrate various behaviours that we observed during the study. Some players find the missing person before retreating, others do not, but this is not relevant to our discussion.

In figure 6.7a, the player has no support except for the beacons. There is no clear search pattern, and they do not appear to follow any particular retreat path and they seem to be lost. Eventually, they get closer to the starting point and follow the beacons for a short way until they reach their destination.

In figure 6.7b, the player also has no support. However, they are able to perform a very tidy search by following the walls and then crossing the central area by following the pillars. At the retreat signal, they return directly to the starting point by the shortest path.

In figure 6.7c, the player has the map interface. They search by following the walls and then crossing through the middle of the car park. At the retreat signal, they initially walk around in a loop before following a reasonably direct path to the starting point. They do not retrace their steps all the way but when they get close to the starting point they leave their previous path; they may have seen the star on the map or the traffic cone on the ground. The behaviour at the start of the retreat phase could have been caused by errors on the map, poor comprehension

of how the map or the direct arrow worked, or simply indecision (i.e., starting to follow the map but then choosing to follow the direct arrow).

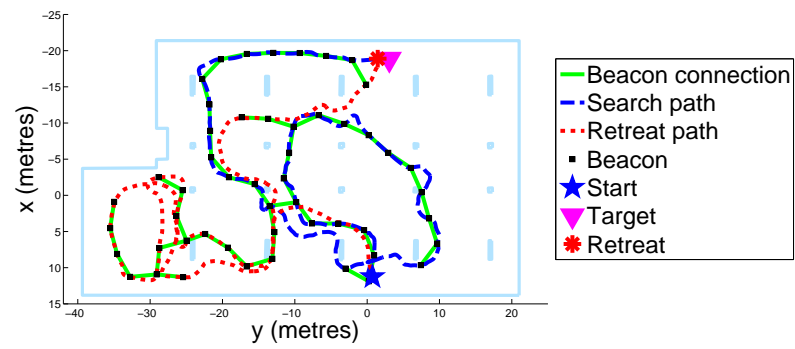
In figure 6.7d, the player has the map interface. There is no clear pattern to their search. When they find the missing person, they retrace their steps until they get close to the starting point. Similar to the previous example, they do not follow their path all the way but they break off from it when it passes close to their destination. This saves them some time.

In figure 6.8a, the player has the arrow interface. They methodically search most of the building but miss a few rooms, including the one where the missing person is located. They retreat by retracing their steps. We can assume that they are following one of the arrows because they overshoot a turning slightly, before coming back a few steps. They finish their retreat by following the corridor rather than going through the room as they did previously which suggests they knew where they were going.

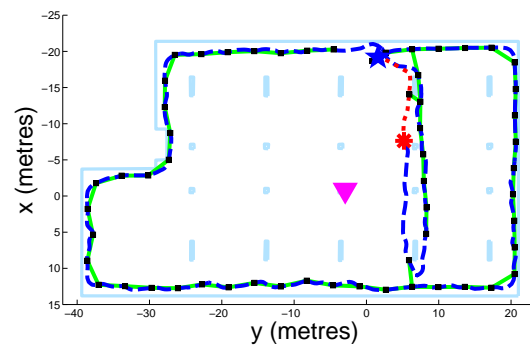
In figure 6.8b, the player has the map interface. They search a large portion of the building but their pattern is not optimal because they cover some areas twice. They do not retrace their steps to retreat. They seem to take a more direct route but they unnecessarily enter several rooms along the way. It is not clear whether they were continuing to search for the missing person, or whether they were misled by the map and thought that the starting point was in one of those rooms. They may have tried to follow the direct arrow which pointed in a straight line to the starting point.

In figure 6.8c, the player has the map interface. This player took the optimal path back to the starting point, without retracing their steps. From this trace, we cannot tell whether they were using the map or whether they remembered exactly where they were.

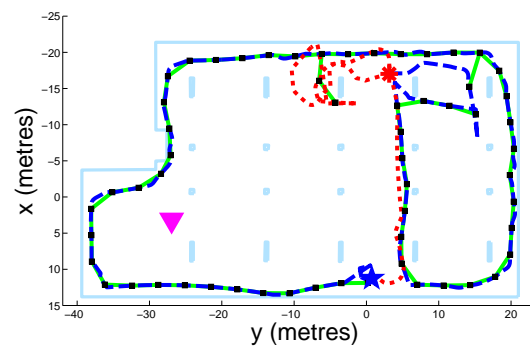
In figure 6.8d, the player has no support except for the beacons. They get lost during their retreat, and seem unable to get out of the left half of the building.



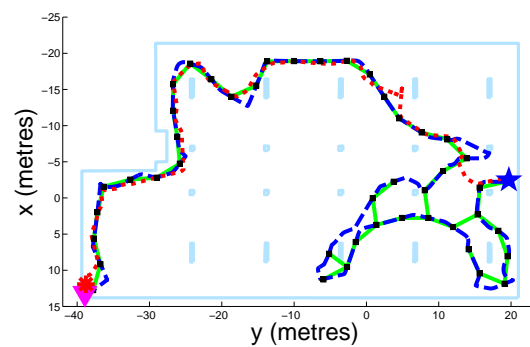
(a) Player 0 – No support.



(b) Player 1 – No support.



(c) Player 7 – With map.



(d) Player 9 – With map.

Figure 6.7: Selection of traces showing different search and retreat behaviours in the car park.

They walk in circles along the same path several times before returning to the exit via an unexplored corridor. After this, they go directly to their starting point, possibly because they see a beacon in front of the door.

6.4 Discussion

The outcome of this study is not as clear as we anticipated, but we believe we have nonetheless learnt valuable lessons about our methods, as well as the navigation problem. In this section, we aim to draw some helpful conclusions regarding the use of a video game as a study tool for navigation systems, and regarding the deployment of our navigation system, albeit in a virtual environment.

6.4.1 Virtual reality as a study tool for sensor-based systems

Working in virtual reality allows us to bypass some of the difficulties associated with real-world sensor-based systems, but it is not always straightforward to transpose a system into a virtual environment.

Limitations

Developing our study with the Unity3D game engine³ presented a number of technical challenges. Video games are designed to look realistic on the surface, but underneath it seems they are based on a number of “tricks” designed to give the appearance of reality while operating in a single flow of execution (thread) at a relatively constant framerate. The lack of support for threads and the fact that many functions are only called once per frame present obvious difficulties when it comes

³<http://unity3d.com> (Accessed 2012.09.25.)

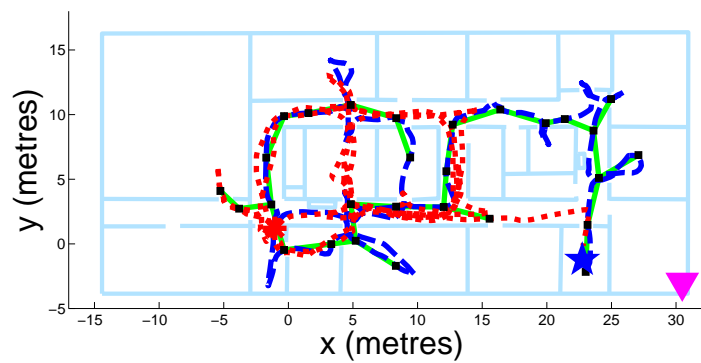
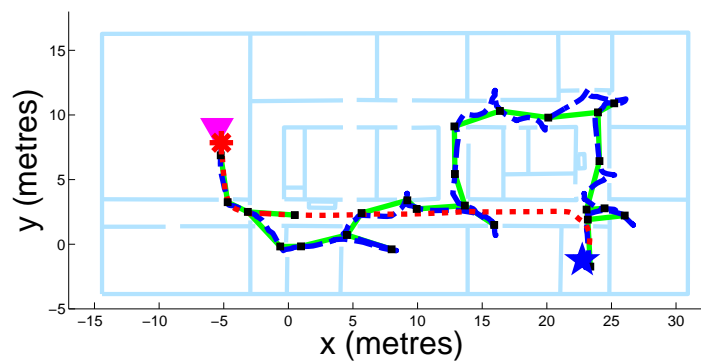
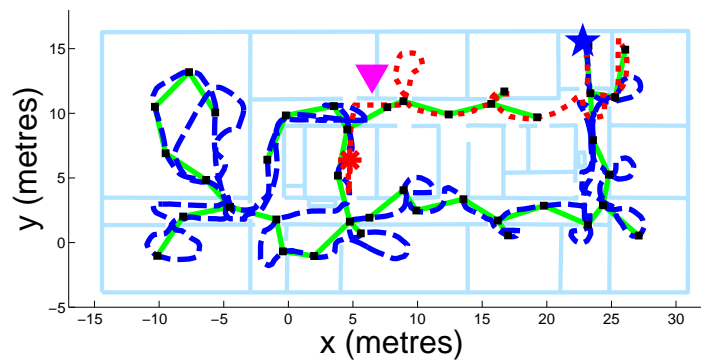
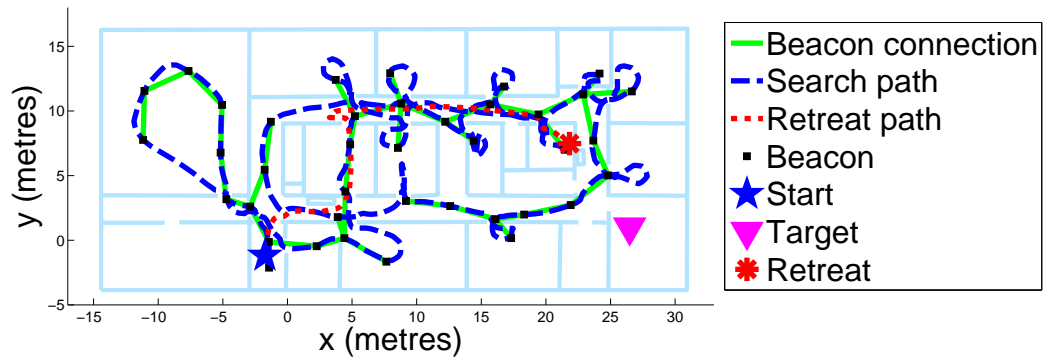


Figure 6.8: Selection of traces showing different search and retreat behaviours in the office building.

to simulating independent sensor nodes or measuring the acceleration of the character's foot. Instead of simulating an inertial PDR system based on accelerometers and gyroscopes, we had to simulate one level higher, step lengths and variations in orientation and direction of travel. We were able to tune our error model so the output resembles typical PDR drift but it takes us one step further away from the system we would use in the real world. More details of the implementation issues are given in appendix B. This awkward coupling between the simulation logic and the graphical presentation is noted by Vala et al. (2009) who have developed the ION framework to simplify the task of creating autonomous agents. Although we do not use agents per se, the sensor nodes and location algorithm could be considered as such, and this framework could provide a convenient solution to some of the issues we faced.

Two participants during our main study felt sick enough to stop playing, and several others during our pre-study. This is a side-effect of playing a first person video game where the screen is animated with a rather jerky walking motion, combined with the low visibility and the lack of landmarks within the game itself. To decrease the chance of motion sickness, we increased the lighting in the study room and opened a window, but this decreased the realism of the game even further. Realism is a concern because we want our players to be fully invested in the game and not distracted by the real world. This is one reason why we asked all players to wear a set of headphones through which they could hear their character breathing and walking. Unity3D allows us to decrease visibility and remove lighting but not to create physically realistic smoke for instance⁴. Nonetheless, although our players did not experience the physical and mental pressure that a firefighter would go through, we observed that several players were taking their missions very seriously. One participant was physically startled when she discovered the body in a corner, another mentioned feeling "panic", and several were reluctant to give up the search without finding the missing person. This suggests that we achieved a

⁴We used the "fog" setting in Unity3D.

level of immersion sufficient to maintain some psychological pressure on the players. This increases the difficulty of the challenge for them and makes it more likely that usability issues will come to light. As far as we know, most virtual reality studies in the literature are more immersive than our study. Riecke et al. (2005) find significant improvements in spatial orientation abilities when the participants have a wider field of view. Although they did not consider a setup such as ours or a low visibility scenario, their results suggest that players may perform worse using our smaller screen than they would in reality. We note, however, that firefighters wearing a breathing mask have a relatively narrow field of view so this may contribute to the difficulty in a way that is beneficial for the study.

User studies take time. In fact, since the video game mission takes place in real-time, it takes as long for the participant as it would with a real world system. The virtual environment saves on development and set up time, as well as equipment costs, but the user study itself takes the same amount of time. For this reason, we were not able to compare different beacon deployment parameters or sensor models (e.g., RF ranging beacons vs ultrasound beacons vs RFID tags vs visual markers). Despite these time requirements, virtual reality and video games may still offer some solutions. If a game is designed to be appealing in itself, it could be released to the wider online community, and results fed back to the researcher automatically. Von Ahn's "games with a purpose" (von Ahn, 2006) follow this principle. Although he has applied it to simple word and image games which work well in a web browser, the idea could maybe be extended to include more elaborate games such as our study.

We only used two environments in our study, the car park and the office building. Each participant explored each environment three times and their search and retreat strategies evolved as they became more familiar with the layout. Although the study was fully counterbalanced with respect to the navigation interface and the environment, although there was no clear learning effect visible in the questionnaire results, and although the players started from a different place each time,

using the same building multiple times was not ideal and may have contributed to masking more interesting effects. We chose to only use these two building layouts because the 3D models were made available to us by Markus Klann and his colleagues from Fraunhofer FIT, and we had neither the time nor the experience to design more. Creating different building layouts need not be too time consuming and it could perhaps be done semi-automatically by adding and removing walls and doors from a grid. However, one challenge would be to create multiple floor-plans with similar characteristics (e.g., room sizes, number of doorways, number of doorways per room). The other challenge would be to maintain a reasonably realistic layout; buildings are rarely designed arbitrarily and information such as the location of stairs and exits can often be guessed. In a sense, it would be easier for the programmer to design worst case environments such as an empty surface with no features at all, or a random maze, rather than strive for realism.

Realism affects another aspect of the game design, namely the position of the missing person. In our study, they were placed in a random location (after the player had searched for some time), but in reality they would be more likely to be near a wall or hidden behind a closed door. Some players assumed the person would be in the same location each time (even though we clearly told them before the study that this was not the case) and others assumed they would be in a similar location (e.g., near a pillar, in a corner, or away from the walls). These assumptions would not be unreasonable in practice but they introduce complex dependencies between conditions beyond the objectives of this study.

Recommendations

For future work, we would recommend defining the context of the study more clearly; either we are evaluating and comparing the algorithms, sensors and general system in a very abstract, controlled, and possibly unrealistic way, or we are testing them in a realistic context to determine whether they provide a solution to a real

problem, and whether they can be used in practice. In the first case, we could very well use an automatically generated maze of a given length with the missing person at one end and the player at the other. In the second case, we would create a replica of an existing building, including furniture, and we would take care to place the player and the missing person in carefully chosen locations. The controlled experiments have the benefit of being easy to design, and could be run many times with less effort, while the realistic experiments are suitable for getting valuable insight from professional firefighters.

The lack of significant results in our numerical data is disappointing. Although most participants claimed verbally that they used the navigation support and that it gave them confidence, or was better than nothing, this was not reflected in their responses to the in-game questionnaire or the quality of their search and retreat traces. We designed the interfaces to be simple, but there was confusion for some people about where the map or arrow were leading them. There is almost certainly a learning curve where the players discover for themselves how the navigation system works and how reliable it is. Results may have been more consistent if we had required players to perform three complete missions in an additional dummy environment rather than just giving them a few minutes to practise.

Another explanation for the inconsistent results could be that participants inexperienced in search and rescue were unable to evaluate their own performance. They may have incorrectly attributed the outcome of their search strategy to luck, or conversely they may have thought that their strategy enabled them to find the exit when in reality they were simply lucky. Several participants claimed to have successfully used certain features of the buildings to assist them, but they did not realise that identical features were present at other locations and that their strategy could have been very misleading. Our in-game questionnaire and interview questions were designed to determine whether the outcome of the mission was due to skill, or luck, or the navigation support system, but it seems that the players

themselves were not necessarily aware of the true cause, due to their lack of previous experience or failure to correctly analyse their performance. It is possible that the researchers who took part in the pre-study simply had more perspective and a better analytical approach than the participants of the main study who were mostly students. We may have had better results if our participants had been trained firefighters who already had well assimilated strategies and expectations of themselves. The root cause of these issues may have been our compromise between a controlled study yielding quantitative results, and a realistic study yielding qualitative feedback from professionals.

6.4.2 Evaluation of the non-infrastructure-based navigation system

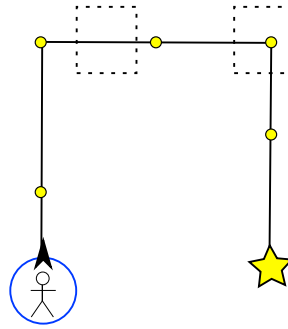
Participants in the study gave us feedback on how they felt about using the different navigation tools. Using this and examining the paths they followed during each mission, we can draw tentative conclusions about our system.

Beacon navigation

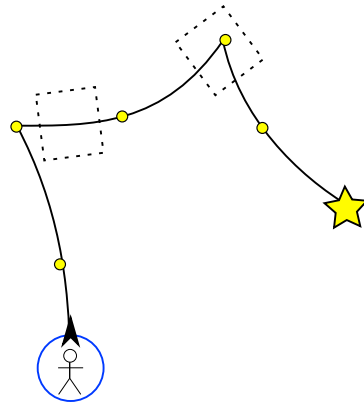
The beacons were by far the most appreciated means of support because of their simplicity. There is no doubt about why they are there and their presence does not claim anything beyond the fact that the player was at this point earlier in the game. This suggests that the preferred solution for some people might be the simple provision of high-visibility colour-coded markers with an automatic deployment system. This solution does not, however, allow many of the advantages that would be available with the full system we have been investigating (e.g., reliable communication channel, alternative exits, shared maps with team members).

Arrow navigation

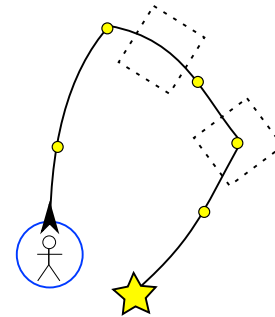
Several people preferred the arrow that pointed directly to the starting point, rather than the more complicated map or the beacon-by-beacon arrow, because it was simple and did not require much interpretation. The direct arrow made shortcuts very easy in the car park because there were no obstacles other than easily avoidable pillars. Some players successfully took shortcuts in the office building. Maybe they remembered the structure of the building enough to avoid walking into a room and having to backtrack. Nevertheless, we have concerns about the direct route arrow because it is sensitive to large scale errors in the map. If the starting point is a long way from the player's current position, the relative error between the two might be large and the arrow might lead the player in the wrong direction with no guarantee that the position estimates will be corrected because they will not necessarily encounter previously deployed beacons (situation described by Fischer et al. (2008)). In contrast, if they follow the beacon-to-beacon arrow or the path on the map, they will be following a trail of beacons on the ground which serves as a reassuring confirmation that they are on track and also provides the measurements required to correct the position estimates (fig. 6.9). Part of the confusion surrounding the beacon-to-beacon arrow may have been due to the parameters used to determine which beacon we should be guiding the player to and when we should guide them to the next one. There is a compromise between trying to guide them along a precise path or a smooth path. On the one hand, a precise path will guide them around obstacles and through doorways, but the arrow will sometimes appear unstable due to the position estimate updates or if the player is unable to follow the path closely enough. On the other hand, a smooth path will be a lot more stable and less subject to the fluctuations of the estimates, but it will sometimes be ambiguous about exactly which door to take for instance, because it is only an approximation of the initial path.



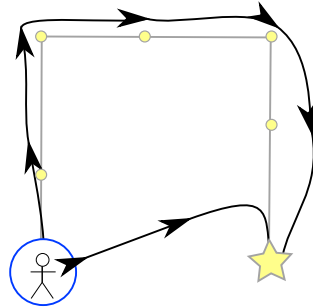
(a) Ground truth.



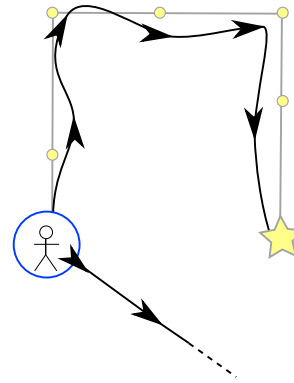
(b) Distorted estimates (ex. 1).



(c) Distorted estimates (ex. 2).



(d) Player's retreat path (ex. 1). They can successfully retrace their steps thanks to the regular position corrections from the beacons. Attempting to reach the exit directly leads them in the wrong direction and they can only correct their path when they intersect previously deployed beacons.



(e) Player's retreat path (ex. 2). They can successfully retrace their steps thanks to the regular position corrections from the beacons. Attempting to reach the exit directly leads them in the wrong direction with no possibility of correction.

Figure 6.9: Players can reliably retrace their steps in the presence of map errors by following indications from the navigation system. They only see a small portion of the map (dotted rectangles) with negligible error at any given time. The beacons provide position corrections to the navigation system as the player walks past them.

Burigat and Chittaro (2007) conducted a navigation study in a virtual environment and found that inexperienced players preferred a three-dimensional arrow rather than a two-dimensional arrow or a radar view (similar to our map but without the connecting lines). They hypothesise that inexperienced players prefer the interface that requires the least mental geometric transformations. This is consistent with the opinions expressed by some of our players. It also suggests that a three-dimensional arrow may be an improvement over the two-dimensional arrow that we implemented. We note, however, that in their study players were able to fly around in three-dimensions instead of being constrained to moving in an approximately two-dimensional plane such as in our simulator.

Map navigation

Errors in the position estimates sometimes cause lines to overlap on the map where they should not and the beacon-to-beacon arrow to spin around erratically. Not only does this cause confusion and prevent people from following instructions correctly, but it also means the system loses their trust. A partial technical solution may be to only display beacons and their corresponding lines on the map if we are certain that they are in close proximity. Proximity can be verified when we have received recent measurements from them. This would allow us to avoid confusion when there is a lot of drift in the map and parts overlap that should not. A simple algorithm would be to only display beacons that we have measured in the past five seconds and beacons that are up to four hops away from them (or another arbitrary hop-count).

The tree structure used for representing the path and beacon layout was only adopted because it simplified certain aspects of the visualisation. Our system could be substantially improved by replacing the tree structure with a better connected directed graph. If the player is within range of several beacons at the same time, we can assume that these beacons are all accessible from each other; in other words, if

the player is near one of the beacons, they can easily walk to the other. When this happens, we could add an edge connecting each of these beacons in our navigation graph. This would open up many more options for finding the shortest (or safest) path to the exit. It would come at the expense of more computation but given the modest size of the graph (a few hundred vertices of degree two or three) it should not present a major challenge even if the shortest path is recomputed frequently.

Player behaviour

One of the goals of our study was to see if players would find a good compromise between following the guidance provided by the system and using their own navigation skills. Some of them achieved this perfectly and compensated for the errors of the system, thus giving it time to correct itself. Others got lost, either by ignoring the navigation support, or by relying upon it entirely and not using any skill of their own. A recurring comment from many of the participants of the pre-study and study is that having a system that you can trust to get you to safety helps you to relax and focus on the task at hand (the search). However, in many cases they seemed not to use the navigation support but it was often unclear why. One person told us she was unable to read a map and another explained he was deliberately ignoring the navigation support so he could refine his own technique. Possibly other people fell into one of these categories as well. In most cases where the participants chose to follow the guidance provided, they reached the starting point quickly. In retrospect, we may have over-emphasised the unreliability of the system. Despite this, some players blindly followed instructions when the system was incorrect, looking for the exit in the same rooms over and over again even though the exit was not there.

6.4.3 Thoughts on using inertial PDR and SLAM

Any navigation interface that orients itself according to the user, like our map, needs to have a reliable estimate of the orientation of the user relative to the direction they are travelling in. For example, if the person is walking forward they expect to see their icon on the map move up (or all the other icons move down), and if they step to the right while still facing forward, they expect to see their icon moving right relative to the other icons on the screen. The key information required to achieve this is not the direction of travel or the orientation, but the difference between the two. This is fortunate because, although inertial foot-mounted PDR drifts and thus does not allow us to measure the absolute direction of travel or orientation reliably, it will give us a fairly accurate estimate of the difference between the two (assuming we know how the inertial sensors are attached to the wearer, e.g., with the x axis pointing forward). We can therefore distinguish between the user walking forwards or backwards, for instance. As long as they continue to face in the same direction, we should not reorient the display of the map, regardless of their direction of travel (e.g., walking backwards, sidestepping). We must take this into account in the implementation of the PDR system. Without drift, a number of variants would give the same result, but because the sensors are imperfect, we need to output length of movement, change in orientation, and difference between orientation and direction of travel (or change in direction of travel and difference between orientation and direction of travel), in order to maintain an orientation of the display consistent with the movements of the user. A useful side effect is that when the SLAM system corrects the position estimates it can also correct the direction of travel (because the two are highly correlated) and the orientation (via the difference). The estimated direction of travel and orientation may still drift but they will drift together and the navigation interface will be consistent with the user's movements.

Due to the nature of SLAM and the lack of anchor points in our current sys-

tem, we cannot tag arbitrary locations, only beacons that we can physically locate thanks to sensors. In the study, the starting point itself (the traffic cone) is not marked on the map, but rather the first beacon is. In principle, even with perfect measurements, the position estimates of the beacons and the player could all drift by a large amount during the course of the mission and yet still be perfectly accurate with respect to each other (and produce perfect navigation support). In other words, the frame of reference in which these entities are tracked could change, and we would not notice it because we are only interested in their positions relative to each other. However, if we tagged a particular location on the map such as the starting point, or an alternative exit, at one point in the game, a few minutes later the frame of reference might have shifted and these coordinates would no longer be relevant. In contrast, sensor nodes have their position estimates updated whenever a measurement is taken. In our implementation, it is only when we have a direct measurement to that particular node; however, in more complex implementations using a full covariance matrix, all beacon estimates could be updated with every measurement from any other beacon. This will be of practical importance for more advanced uses of our navigation system because any area of interest to be highlighted on the map must be attached (physically or virtually) to a sensor node.

6.5 Conclusion

Using a video game, we have demonstrated how to build and evaluate a navigation system based on inertial dead reckoning and sensor nodes with the ability to perform ranging measurements. This system provides navigation support in unknown environments with no existing localisation infrastructure. The game generates sensor data and position estimates with similar characteristics to those we have observed with real sensors; in other words, the in-game measurements contain

realistic errors. The navigation support is built around the simultaneous localisation and mapping (SLAM) algorithm from the previous chapter. The algorithm is simple but exposes some of the issues inherent to this type of system, particularly drift and map inconsistencies.

During the development, we noted certain practical issues that would warrant more consideration in future work. For instance, we faced implementation challenges as we tried to make our in-game sensors as realistic as possible and realised that the game engine makes this task difficult. We also discovered that inertial movement estimates need to be processed in a certain way to avoid inconsistencies in the displayed map.

The study did not yield the clear results we had hoped for, but the process has helped us to better see which aspects of it should be addressed. We have realised the importance of giving participants more time to rehearse their search and retreat strategies before starting the evaluated study. We also acknowledge that there would be considerable benefits in using people with search and rescue experience as study participants because their prior experience would give them perspective and allow them to more reliably evaluate their own performance.

On the whole, the feedback from our participants was positive. There was sometimes confusion about how to interpret the map or the movements of the navigation arrow, but other participants were able to use them well. While we observed some people following navigation support too carefully and apparently not considering where their path was taking them, others were easily able to include the provided information into their own strategy. The “electronic breadcrumb” beacons were mentioned most positively, highlighting the importance of keeping a navigation system grounded in physical reality.

We suggest that future studies of this kind should be either designed as controlled experiments to yield solid quantitative data regarding the algorithms and

interface, or as realistic scenarios which provide professionals with the opportunity to give qualitative feedback on the practical deployment and use of the system. Our attempt to combine these two options into a single study may have jeopardised the overall outcome. Running two distinct types of study might also make evaluation of the results easier. We struggled to find good metrics for judging the performance of the players and eventually resorted to manually rating the recorded traces. A very controlled study could make it more straightforward to count mistakes, or automatically measure the time taken or the time lost.

The navigation capabilities of the system do not require any anchor points with known positions. The lack of anchor points only means that the position estimates cannot be simply overlaid on a floorplan or satellite image. When such points are available via GPS receivers or manual placement of nodes at known locations, the system gains additional capabilities and can be used by a third party, such as the incident commander, to track people in a building.

We are satisfied with the design of the navigation interface. Although some users found some aspects confusing, we believe this can be resolved through better training and experience. We were initially concerned that the interface would be either too simplistic and that users would want more information, or that it would be too complex and overwhelm them with information. Preferences were divided between the arrow interface and the map, but we seem to have struck the right balance in terms of the amount of information provided and the level of detail.

Our navigation system can certainly be improved upon, but we believe the principles are right. The SLAM algorithm can be made more robust and effective by using one which maintains more cross-correlations than we do (Julier and Uhlmann, 2006). Alternatively, a system based on a spring model (Golfarelli et al., 1998) could also provide a way to re-evaluate the entire map and make it consistent when the user completes a loop. Independently from the SLAM algorithm, the part of the system that computes the navigation map and the direction of the arrow should

be redesigned to use a more flexible data structure than a tree, so it can include additional connections between nearby beacons and provide more and better paths. These improvements are all subject to computational limitations, especially in the virtual environment where framerate dictates certain restrictions, but also in a real world implementation where communication and power resources are limited.

The choice of sensors for the beacons remains an issue. One of the reasons we conducted our study in a virtual environment was the lack of clear solution for the beacons. We have had some success with ultrasound but also many problems, and radio-based ranging technologies do not seem to provide the required accuracy. This is the major factor that requires further work and would either make our system a reality or cause it to be abandoned.

We have taken the development of a navigation system further towards real-world implementation than research projects typically do, by testing not only the algorithms but also the user interface and the effect of the users' behaviour on the system. We have thus been able to discuss issues that usually only present themselves much later in the design and production process. By developing the system in a virtual environment, we have highlighted some of the issues that designers of navigation support systems may face, despite not having access to the ideal technology for the beacons. We believe a virtual environment such as a 3D game engine, with all its limitations, will allow a developer to go through several system design, implementation, and evaluation iterations, and enable them to improve the final navigation system at lower cost.

Conclusion

At the start of this thesis, we noted that amongst all the research into localisation and location-aware technologies there was very little that was applicable to unknown and uninstrumented environments. We were concerned by the fact that many of the proposed solutions were highly dependent on existing infrastructure or prior knowledge, and were thus not robust or flexible enough to support important tasks such as firefighting or search and rescue.

7.1 Contributions

We have given an overview of the literature concerning localisation from many different fields, and have shown that, despite the many advances in technologies and algorithms, and despite the development of specialised systems which provide reliable solutions to many problems, there is no given solution which provides localisation for the type of situations encountered by emergency responders. More and more researchers are acknowledging this as a worthwhile application, and are investigating further in this direction, but our survey of this particular area of research revealed that the community has not yet found a satisfactory solution

and that all proposals so far fall short of the requirements in terms of robustness and being entirely self-contained.

One of the key difficulties is getting away from the idea that we need to provide location estimates in an absolute coordinate frame. We have tackled this by describing practical applications that only require position estimates relative to a few nearby points rather than a large scale picture. We also introduced a scalable algorithm based on peer-to-peer measurements between sensor nodes that could enable these applications to be supported robustly, even in the case where all sensors are mobile.

Our experience with ultrasonic sensor nodes and our localisation algorithm, with their practical limitations, prompted us to explore a separate strategy for tracking pedestrian movement using inertial sensors: inertial pedestrian dead-reckoning. This method is the foundation of many navigation systems aimed at firefighters due to its self-contained nature but there are no clear implementation details available to the research community, resulting in many researchers using sub-optimal algorithms or wasting time redeveloping them from scratch. We provided a reference implementation and information on the required hardware, along with details of how to make the most of this tracking method, emphasising its strengths and limitations.

In order to mitigate the flaws of both previous techniques, we combined the inertial pedestrian dead-reckoning algorithm with the sensor nodes using a concept commonly used in robotics: simultaneous localisation and mapping. We demonstrated how this method could be applied to pedestrians, and evaluated its practical implementation, followed by a discussion of its application in a real setting.

Finally, we evaluated this system by implementing the localisation and mapping algorithm, and a navigation system based upon it in a virtual environment. This allowed us to highlight some conceptual challenges in the design and implementation of this type of system, as well as discuss comments from participants in

a usability study. We made several suggestions concerning the evaluation methodology for navigation systems such as ours, as our system has specific characteristics different from ones previously studied in the literature.

7.2 Challenges and lessons learnt

During the course of our work, several challenges have been recurring and have hindered our progress. Although we have been able to work around or solve them in certain instances, they warrant further investigation.

Sensor hardware has limited our work in some respects. When we started we were not aware of the exact requirements, but we now have a clearer idea of which sensors will give acceptable results. This is true especially of the inertial sensors. The question of which technology to use for the beacons remains unclear. We used ultrasound in our studies primarily because this is what we had experience with and what was available at the time. In retrospect, after having wrestled with obsolete radio modules and simplistic signal processing, we still believe that ultrasound offers the most promising characteristics due to its accuracy, its ability to produce angular measurements, and the fact that it requires line of sight. We hope that others will see the value of improving this technology. Radio ranging has made some progress since we started this work and may soon offer a viable alternative for range and bearing measurements in indoor environments.

The evaluation of the navigation systems we have designed poses both practical and conceptual difficulties. From a practical point of view, it is difficult to obtain continuous high-resolution ground truth over large areas, especially indoors. From a conceptual point of view, even having obtained the ground truth, evaluating the quality of the location estimates is not an obvious task because we do not use a fixed coordinate system. The notion of localisation quality itself becomes more

complex once we start evaluating navigation systems, because we need to look at location estimates relative to the person using the system rather than from an outside point of view. With hindsight, it seems necessary to us to separate two types of system evaluation: one which will generate the quantitative data needed to compare the effects of different parameters, another which will yield qualitative feedback from professional end users. The first type of study needs to be repeated many times, and would test the system in a very controlled, and therefore abstract and unrealistic environment. The second type of study would be conducted only a few times in a very realistic environment with a very targeted set of users.

7.3 Vision for the future of navigation support for emergency response

This work has inspired us to see a future where emergency responders have access to reliable tracking and navigation support in all environments. As we have found out more about the conditions they face and the physical and mental challenges, we have realised that this is a worthwhile area of research, and one that will be truly appreciated when it starts offering mature solutions. We have also seen that working solutions are within reach. Although we have emphasised the need for a general solution to the localisation problem, certain aspects of emergency response actually make the design easier. For instance, we do not necessarily need to have an algorithm that can deal with tens of thousands of sensors if there are only going to be a few hundred deployed in any given scenario. Similarly, since most search and rescue missions by firefighters in low visibility will take place using breathing apparatus, the duration of these missions is limited to roughly 30 minutes, which means that batteries and algorithm reliability do not need to last much longer.

We can easily envision the following scenario. A team of firefighters enter a smoke-filled building to rescue some people trapped inside. As they progress in

complete darkness, they use their head-mounted display to keep track of which parts of the building they have searched and where their companions are. Although the display cannot tell them anything about the layout of the building, it allows them to focus on the search without worrying about how to get to the exit or how much time they have left before their air runs low. If they need to reach the exit quickly, they simply follow the arrows on the display to retrace their steps. The arrows and the path traced on the display are always oriented according to the direction the wearer is facing thanks to information provided by an inertial measurement unit built into their helmet. When their paths intersect with that of another team which entered via another entrance a few minutes earlier, the system recomputes the shortest path to the alternative exit and will guide them to whichever one is closest.

As they advance, they deploy tiny beacons in key locations such as doorways or at corners. These beacons transmit ultrasonic and radio ranging signals which allow the navigation system to keep track of their path. However, the system can continue to work reliably even if the beacons are destroyed or moved thanks to the inertial sensors embedded in their boots. The beacons also act as radio relays to guarantee a reliable voice and data communication link to the command post outside. They have a third purpose as visual markers thanks to their flashing ultrabright LEDs and reflective surface. The inertial tracking system functions for any type of pedestrian motion, including walking, crawling, running, or climbing stairs and ladders, and it is made more accurate by the combination of data from different sources. Each firefighter has one sensor on each boot, and they work in teams following approximately the same path. The tracking algorithm uses the inertial data from each sensor as well as the ranging data between team members, or between team members and beacons, to produce the most reliable position estimates. As the team follow the walls inside the building according to standard practice, their estimated positions are transmitted to the command post where the commander starts to see a clear outline of the internal structure, including rooms,

corridors, and stairs. Although a floorplan is not required, if one becomes available, it only requires a few swipes across the screen for the commander to align the paths with it.

If one team needs assistance, another team can simply follow the navigation instructions to find them. They can also easily spot areas that the initial team has not searched. None of these features requires a floorplan or any external location system. However, if one of the firefighters is equipped with a GPS receiver, the system can use this information to further improve its estimates and to align the floorplan or satellite view automatically. The navigation system works reliably for a single person with only a minimal number of sensors but its accuracy and reliability increase as the number of users increases and additional sensors or data sources are connected. It is also completely autonomous and capable of guiding the wearer without any connection to the outside, and yet it allows effective collaboration between team members and between teams. We find this picture to be inspiring, even more so as we can see relatively clearly how all the features could realistically be developed.

There is an increasing amount of research in the area of pedestrian navigation in uninstrumented environments and we assume that the vision we described is a shared one. More and more companies are advertising tracking solutions for GPS-denied environments using technologies covered in this thesis, while others are actively searching for solutions that will allow them to complete their software packages with indoor location information. Calls for research proposals are also being made in this area. We are encouraged to see that this is being acknowledged as a relevant area of research and commercial development. We are confident that the community has the ability to successfully tackle this problem in order to enhance the safety of emergency responders, and provide convenient, robust, and low-cost navigation and tracking solutions for a range of users, without the infrastructure and prior knowledge constraints of existing systems.

Inertial pedestrian dead reckoning implementation in Matlab

This appendix contains the implementation of our foot-mounted inertial pedestrian dead reckoning algorithm in Matlab script.

Listing A.1: Foot-mounted inertial pedestrian dead-reckoning Matlab code

```
% Inertial pedestrian tracking.
%
% For best results use a foot-mounted inertial measurement unit with an
% accelerometer range greater than 10g and a gyroscope range greater than
% 900 degrees per second and at least 50 samples per second. The IMU does
% not need to estimate orientations.
%
%
% Copyright December 2010, Lancaster University.
% Authors: Carl Fischer, Poorna Talkad Sukumar.
% http://eis.comp.lancs.ac.uk/pdr/

clear all;

%% Read data from file.
% Data should include timestamps (seconds), 3 axis accelerations (m/s^2), 3
% axis gyroscopic rates of turn (rad/s).
data = importdata('1.csv'); gyro_bias = [-0.0156 -0.0101 -0.0020]';
%data = importdata('2.csv'); gyro_bias = [0.0066 -0.0071 0.0120]';
```

```

%data = importdata('3.csv'); gyro_bias = [0.0066 -0.0071 0.0235]';
%data = importdata('4.csv'); gyro_bias = [0.0066 -0.0071 0.025]';
%data = importdata('running.csv'); %gyro_bias = [0.0386 -0.0488 -0.00]';
data_size = length(data.data);
timestamp = data.data(:,1)'; % Timestamps of measurements.
acc_s = data.data(:,2:4)'; % Accelerations in sensor frame.
gyro_s = data.data(:,5:7)'; % Rates of turn in sensor frame.
g = 9.8; % Gravity.

%% Initialise parameters.
% Orientation from accelerometers. Sensor is assumed to be stationary.
pitch = -asin(acc_s(1,1)/g);
roll = atan(acc_s(2,1)/acc_s(3,1));
yaw = 0;

C = [cos(pitch)*cos(yaw) (sin(roll)*sin(pitch)*cos(yaw))-(cos(roll)*sin(yaw)) (
    cos(roll)*sin(pitch)*cos(yaw))+(sin(roll)*sin(yaw));
    cos(pitch)*sin(yaw) (sin(roll)*sin(pitch)*sin(yaw))+(cos(roll)*cos(yaw)) (cos(
    roll)*sin(pitch)*sin(yaw))-(sin(roll)*cos(yaw));
    -sin(pitch) sin(roll)*cos(pitch) cos(roll)*cos(pitch)];
C_prev = C;

% Preallocate storage for heading estimate. Different from direction of
% travel, the heading indicates the direction that the sensor, and therefore
% the pedestrian, is facing.
heading = nan(1, data_size);
heading(1) = yaw;

% Gyroscope bias, to be determined for each sensor.
% -- Defined above so we don't forget to change for each dataset. --

% Preallocate storage for accelerations in navigation frame.
acc_n = nan(3, data_size);
acc_n(:,1) = C*acc_s(:,1);

% Preallocate storage for velocity (in navigation frame).
% Initial velocity assumed to be zero.
vel_n = nan(3, data_size);
vel_n(:,1) = [0 0 0]';

% Preallocate storage for position (in navigation frame).
% Initial position arbitrarily set to the origin.
pos_n = nan(3, data_size);
pos_n(:,1) = [0 0 0]';

% Preallocate storage for distance travelled used for altitude plots.
distance = nan(1,data_size-1);
distance(1) = 0;

```

```

% Error covariance matrix.
P = zeros(9);

% Process noise parameter, gyroscope and accelerometer noise.
sigma_omega = 1e-2; sigma_a = 1e-2;

% ZUPT measurement matrix.
H = [zeros(3) zeros(3) eye(3)];

% ZUPT measurement noise covariance matrix.
sigma_v = 1e-2;
R = diag([sigma_v sigma_v sigma_v]).^2;

% Gyroscope stance phase detection threshold.
gyro_threshold = 0.6;

%% Main Loop
for t = 2:data_size
    %%% Start INS (transformation, double integration) %%%
    dt = timestamp(t) - timestamp(t-1);

    % Remove bias from gyro measurements.
    gyro_sl = gyro_s(:,t) - gyro_bias;

    % Skew-symmetric matrix for angular rates
    ang_rate_matrix = [0 -gyro_sl(3) gyro_sl(2);
        gyro_sl(3) 0 -gyro_sl(1);
        -gyro_sl(2) gyro_sl(1) 0];

    % orientation estimation
    C = C_prev*(2*eye(3)+(ang_rate_matrix*dt))/(2*eye(3)-(ang_rate_matrix*dt));

    % Transforming the acceleration from sensor frame to navigation frame.
    acc_n(:,t) = 0.5*(C + C_prev)*acc_s(:,t);

    % Velocity and position estimation using trapeze integration.
    vel_n(:,t) = vel_n(:,t-1) + ((acc_n(:,t) - [0; 0; g] )+(acc_n(:,t-1) - [0; 0; g]))*dt/2;
    pos_n(:,t) = pos_n(:,t-1) + (vel_n(:,t) + vel_n(:,t-1))*dt/2;

    % Skew-symmetric cross-product operator matrix formed from the n-frame accelerations.
    S = [0 -acc_n(3,t) acc_n(2,t);
        acc_n(3,t) 0 -acc_n(1,t);
        -acc_n(2,t) acc_n(1,t) 0];

    % State transition matrix.
    F = [eye(3) zeros(3,3) zeros(3,3);
        zeros(3,3) eye(3) dt*eye(3);
        -dt*S zeros(3,3) eye(3) ];

```

```

% Compute the process noise covariance Q.
Q = diag([sigma_omega sigma_omega sigma_omega 0 0 0 sigma_a sigma_a
sigma_a]*dt).^2;

% Propagate the error covariance matrix.
P = F*P*F' + Q;
%%% End INS %%%

% Stance phase detection and zero-velocity updates.
if norm(gyro_s(:,t)) < gyro_threshold
    %%% Start Kalman filter zero-velocity update %%%
    % Kalman gain.
    K = (P*(H)')/((H)*P*(H)' + R);

    % Update the filter state.
    delta_x = K*vel_n(:,t);

    % Update the error covariance matrix.
    % P = (eye(9) - K*(H)) * P * (eye(9) - K*(H))' + K*R*K'; % Joseph
    % form to guarantee symmetry and positive-definiteness.
    P = (eye(9) - K*H)*P; % Simplified covariance update found in most books.

    % Extract errors from the KF state.
    attitude_error = delta_x(1:3);
    pos_error = delta_x(4:6);
    vel_error = delta_x(7:9);
    %%% End Kalman filter zero-velocity update %%%

    %%% Apply corrections to INS estimates. %%%
    % Skew-symmetric matrix for small angles to correct orientation.
    ang_matrix = -[0 -attitude_error(3,1) attitude_error(2,1);
attitude_error(3,1) 0 -attitude_error(1,1);
-attitude_error(2,1) attitude_error(1,1) 0];

    % Correct orientation.
    C = (2*eye(3)+(ang_matrix))/(2*eye(3)-(ang_matrix))*C;

    % Correct position and velocity based on Kalman error estimates.
    vel_n(:,t)=vel_n(:,t)-vel_error;
    pos_n(:,t)=pos_n(:,t)-pos_error;
end
heading(t) = atan2(C(2,1), C(1,1)); % Estimate and save the yaw of the sensor
(different from the direction of travel). Unused here but potentially useful for
orienting a GUI correctly.
C_prev = C; % Save orientation estimate, required at start of main loop.

% Compute horizontal distance.
distance(1,t) = distance(1,t-1) + sqrt((pos_n(1,t)-pos_n(1,t-1))^2 + (pos_n(2,
t)-pos_n(2,t-1))^2);
end

```

```

%% Rotate position estimates and plot.
figure;
box on;
hold on;
angle = 180; % Rotation angle required to achieve an aesthetic alignment of the
             figure.
rotation_matrix = [cosd(angle) -sind(angle);
                  sind(angle) cosd(angle)];
pos_r = zeros(2,data_size);
for idx = 1:data_size
    pos_r(:,idx) = rotation_matrix*[pos_n(1,idx) pos_n(2,idx)];
end
plot(pos_r(1,:),pos_r(2,:), 'LineWidth',2,'Color','r');
start = plot(pos_r(1,1),pos_r(2,1),'Marker','^','LineWidth',2,'LineStyle','none');
stop = plot(pos_r(1,end),pos_r(2,end),'Marker','o','LineWidth',2,'LineStyle','none');

xlabel('x_(m)');
ylabel('y_(m)');
title('Estimated_2D_path');
legend([start;stop],'Start','End');
axis equal;
grid;
hold off;

%% Plot altitude estimates.
figure;
box on;
hold on;
plot(distance,pos_n(3,:), 'Linewidth',2, 'Color','b');
xlabel('Distance_Travelled_(m)');
ylabel('z_(m)');
title('Estimated_altitude');
grid;

% Display lines representing true altitudes of each floor.
floor_colour = [0 0.5 0]; % Colour for lines representing floors.
floor_heights = [0 3.6 7.2 10.8]; % Altitude of each floor measured from the ground
floor.
floor_names = {'A' 'B' 'C' 'D'};
lim = xlim;
for floor_idx = 1:length(floor_heights)
    line(lim, [floor_heights(floor_idx) floor_heights(floor_idx)], 'LineWidth', 2, '
        LineStyle', '--', 'Color', floor_colour);
end
ax1=gca; % Save handle to main axes.
axes('YAxisLocation','right','Color','none','YTickLabel', floor_names, 'YTick',
    floor_heights,'XTickLabel', {});
ylim(ylim(ax1));

```

```
ylabel('Floor');  
hold off;
```

Navigation algorithm and implementation issues

In this appendix, we give the algorithm used to compute the navigation tree used for the virtual reality study of chapter 6. The details of the inertial PDR algorithm are the focus of chapter 4 (see appendix A for the Matlab implementation), and the SLAM algorithm is described in chapter 5.

The game engine is configured to generate PDR movement measurements every 10 milliseconds, and range measurements to any beacons within range every 30 milliseconds, but in practice this is limited by the framerate of the game (approximately 30 frames per second). We use the physics engine to keep track of which beacons are within a five metre radius of the player, and we use ray casting to determine which of those are within line of sight before generating range measurements. Using the physics engine is a necessary optimisation which avoids us having to iterate over every beacon in the game at every frame. In a real world implementation, nodes that are far apart cannot sense each other and do not generate measurements, so this would not be an issue. If there are no range measurements less than four metres, a new beacon is deployed. Beacons are dropped no more than once per second. We are unable to generate true inertial measurements in the game

due to an approximate character animation, so we instead compute position and rotation differences to simulate a PDR system which gives us step lengths, changes in orientation, and offset between orientation and direction of travel. By adding noise with non-zero mean to these values, we obtain position estimates which resemble PDR with drift. The movement and range measurements are passed to the navigation system itself which is partially described by listing B.1. In particular, it illustrates how the navigation tree is constructed.

Listing B.1: Navigation tree pseudo-Java code.

```

class Beacon {
    Coordinate position;
    Beacon previousBeacon;
    List<List<Beacon>> branches;
}

class Navigator {
    ...
    Coordinate position; // Estimated position of the pedestrian.
    Beacon rootBeacon = null; // Root of the tree = starting point.
    Beacon connectedBeacon = null; // Beacon that the player is currently connected to.
    Beacon targetBeacon = null; // Beacon that the player is being guided towards.
    float nearThreshold = 1.0; // Closer than this distance, the player is considered too close to a beacon and we should guide them to the next one.

    AddBeacon(Beacon beacon) {
        if (rootBeacon == null) {
            rootBeacon = beacon; // First beacon to be dropped is assumed to be the root of the tree.
        }
        if (connectedBeacon == null) {
            connectedBeacon = beacon;
        }
        if (connectedBeacon.branches.count == 0 || // Beacon has just been deployed.
            connectedBeacon.branches.count > 1 || // Beacon is a junction node with multiple children.
            connectedBeacon.branches.count == 1 && connectedBeacon.branches[0].last != connectedBeacon) // Beacon is in the middle of a branch.
        {
            List<Beacon> branch = new List<Beacon>(); // Create a new branch.
            branch.Add(connectedBeacon); // Add the parent node.
            branch.Add(beacon); // Add the current node as the first child.
            connectedBeacon.branches.Add(branch); // Add the new branch to the previous beacon.
            beacon.branches.Add(branch); // Add the new branch to the new beacon.
        }
    }
}

```

```

else // Beacon is the last node of a simple branch.
{
    connectedBeacon.branches[0].Add(beacon); // Append this beacon to the
        end of the current branch.
    beacon.branches.Add(connectedBeacon.branches[0]);
}
beacon.previousBeacon = connectedBeacon; // Keep a link to the parent node
    so we can find our way back later.

ProcessPDRMeasurement(PDRMeasurement m) {
    ... // Update pedestrian position estimate.
}

ProcessRangeMeasurement(RangeMeasurement m, Beacon beacon) {
    ... // Correct pedestrian and beacon position estimates.

    EvaluateConnectedBeacon(beacon); // Check which is the closest beacon.
}

// Find the closest beacon to the pedestrian.
EvaluateConnectedBeacon(Beacon beacon) {
    if (connectedBeacon == null) { // If the pedestrian was not connected to a
        beacon, use this one.
        lastConnectedBeacon = beacon;
    } else {
        float thisDistance = Distance(beacon.position, position); // Distance to
            this beacon.
        float oldDistance = Vector3.Distance(connectedBeacon.position,
            position); // Distance to current "last connected" beacon.
        if (thisDistance < oldDistance) { // If this beacon is closer to our
            current position than the previous one, then connect to this one.
            connectedBeacon = beacon;
        }
        FindTargetBeacon(); // We may be connected to a different beacon, so
            re-evaluate the target beacon too.
    }
}

// Check which beacon we should be guiding the pedestrian towards.
FindTargetBeacon() {
    if (targetBeacon == null) {
        targetBeacon = connectedBeacon;
    }
    if (targetBeacon != connectedBeacon || targetBeacon != connectedBeacon.
        previousBeacon) // We should only be guiding the player towards the
        beacon they are connected to or the previous one.
    {
        targetBeacon = connectedBeacon;
    }

    // Dot product telling us whether the player is 'between' the target and

```

```

        previous target beacons ( $>0$ ) or 'beyond' the target beacon ( $<0$ ).
float prod = DotProduct(connectedBeacon.previousBeacon.position -
        connectedBeacon.position, position - connectedBeacon.position);

float distance = Vector3.Distance(position, targetBeacon.position);
if (distance < nearThreshold || prod > 0)
    targetBeacon = targetBeacon.previousBeacon;
    }
}
}

```

The Unity3D game engine¹ presented some challenges to our work. Although it includes a physics engine which aims to reproduce the effects of physical interactions between objects, it is designed to mimic these effects in appearance only. For instance, it does not allow us to easily simulate independent sensor nodes running their own software, or inertial sensors measuring the accelerations and rates of turn of the pedestrian's feet 100 times per second. One reason for this is that games (at least those created with Unity3D) are designed to run as a single thread, with functions being called at most once per frame. Ideally, our simulator needs to generate measurements, add noise to them, and process them several times per frame. This was not possible, so we adjusted the simulated measurements to mimic the effects of inertial tracking at a lower rate than what we would have in a real world implementation. Our SLAM algorithm from chapter 5 runs as a separate thread but receives the measurements and updates the position estimates via two synchronised (thread-safe) queues in order to not interfere with the main game engine process. Once again, this would not be such a problem with a real system because the measurements would be simply *measured* rather than *generated*, the CPU and GPU would not be busy rendering a 3D landscape, and we would presumably be working in a more thread-friendly environment.

¹<http://unity3d.com/> (Accessed 2012.09.24.)

User study: navigation in a virtual environment

C.1 Study design

Table C.1 gives the order of conditions for the participants of the navigation study. The order in which the environments (car park or offices) were presented was alternated, and the order of the interfaces was balanced. The pre-study was conducted using six participants (numbered 0 to 5) and the full study was conducted using twelve participants (numbered 0 to 11). In retrospect, we should have inverted the order of the environments for participants 6 to 11, but we do not believe this had any effect on the results.

C.2 Verbal questionnaire

These questions took the form of a semi-structured interview based on the points below.

Table C.1: Navigation study design.

Participant	Mission					
	1	2	3	4	5	6
0	cn	ca	cm	oa	on	om
1	on	om	oa	ca	cm	cn
2	ca	cn	cm	om	on	oa
3	oa	om	on	cm	ca	cn
4	cm	cn	ca	on	oa	om
5	om	oa	on	cn	cm	ca
6	cn	ca	cm	oa	on	om
7	on	om	oa	ca	cm	cn
8	ca	cn	cm	om	on	oa
9	oa	om	on	cm	ca	cn
10	cm	cn	ca	on	oa	om
11	om	oa	on	cn	cm	ca

Environment – c: car park, o: offices.

Interface – n: no support, a: arrow, m: map.

C.2.1 Following the three missions in each building

- First impressions.
- Describe your search strategy.
- How effective do you think your search strategies were?
- Describe your retreat strategy.
- Did you feel disoriented at any time during the missions?

In each case, prompt for differences between scenarios and interfaces.

C.2.2 At the end of the study

- Did the two types of building cause you to behave differently?

C.3 Demographics questionnaire

The participant was asked to respond to this questionnaire on a paper form.

- Gender. [Male — Female]
- Age.
- Experience with computer games similar to the study (first person view). [Regular player — Occasional player — Played in the past — Never played]
- Experience with other 3D virtual environments (e.g., modelling tools, flight simulators). [Experienced — Occasional user — Used a few times — Never used]
- General navigation skills (e.g., map reading, finding your way in new buildings or cities). [Excellent — Good — Average — Poor — Awful]
- How do you feel about finding your way in the dark? [Confident — Cautious — Likely to get lost or fall — Terrified — Don't know]

C.4 Summary of interviews

Table C.2 provides a summary of the key points raised during the interviews.

Table C.2: User feedback.

Player	Car park	Office	Beacons Car park	Beacons Office	Arrow Car Park	Arrow Office	Map Car Park	Map Office	other
0	Faster to search. Used D signs.	Easier. Used landmarks.	Used to know where had been.	Used to know where had been.	Easier to retreat; confusing.		Easier to retreat.		
1	Used walls and pillars. Used D signs. More difficult.	Found landmarks. Easier.	Used to know where had been; Used for orientation.		Confusing (instability).	Confirm location.		Confirm location.	Navigation support for retreat only.
2	Followed wall.	Felt panic. Used landmarks.	Not helpful.	Used to know where had been.	Remember where came from (orientation); allowed focus on search; easier to follow in panic.	Used on retreat but got lost.	Remember where came from (orientation); more useful in open space.	Confusing (lines cross).	
3	Difficult to compartmentalise. Used walls.		Used to know where had been; useful.	Not helpful.	Used on retreat.	Used on retreat.	Used on retreat.	Used on retreat.	Navigation support for retreat.
4	Used pillars.	Easier. More landmarks.					Preferred. Useful for retreat.	Useful for retreat.	
5	Rejected D signs. Difficult to search. No landmarks. Darker.	Used fire escape signs.			Frustrating (instability).	Useful for retreat; used to know where had been.	Helpful; useful.	Useful for retreat; used to know where had been.	
6	Confusion.	Easier to search.			Easier than map.	Confusing.		Used to know where had been.	
7	Difficult to search. No landmarks.	Less disorienting. More light.	Used for orientation.	Not helpful.	Liked the arrow.	Easier than map; preferred arrow.			Tools helped navigate and gave confidence; used them.
8	Rejected D signs.	Found landmarks. Not as easy as expected.	Used when nothing else.		Helped when nothing else; sometimes disorienting.	Green arrow didn't move as expected.	Used to know where had been; preferred; helpful on retreat.	Used on retreat.	
9	Difficult.	Easier.	Helpful but not easy	Used (relied).	Confusing (didn't understand).		Cannot use.	Didn't use.	
10	Followed wall. More difficult.	Followed wall.	Used for retreat; dropped near start point.	Adapted to.	Helpful on retreat.		Helpful on retreat.	Adapted away from.	Confusing with no support; assistance made retreat easier.
11	Followed wall. Difficult to search but easier to retreat.	Only a few landmarks.	Used to know where had been.	Used	Helpful for retreat.	Helpful for orientation (retreat); improved search.	Not helpful on search.	Helpful, difficult without.	Navigation support helped focus on search.

Bibliography

- Hénoc Agbota. *Understanding and Mitigating Contention in Wireless Sensor Networks*. PhD thesis, Lancaster University, 2009. 68
- Andreas M. Ali, Kung Yao, Travis C. Collier, Charles E. Taylor, Daniel T. Blumstein, and Lewis Girod. An empirical study of collaborative acoustic source localization. In *Proc. of IPSN*, pages 41–50, Cambridge, MA, USA, 2007. doi: 10.1145/1236360.1236367. 12
- Vincent Amendolare, David Cyganski, R. James Duckworth, Sergey Makarov, Jack Coyne, Hauke Daempfling, and Benjamin Woodacre. WPI Precision Personnel Location System: Inertial Navigation Supplementation. In *Position Location And Navigation Symposium*, Monterey, CA, 2008. IEEE. 34
- Isaac Amundson, Xenofon Koutsoukos, and Janos Sallai. Mobile Sensor Localization and Navigation using RF Doppler Shifts. In *Proc. of MELT*, San Francisco, CA, 2008. 52, 55
- John R. Anderson. Abandoned Cold Storage Warehouse Multi-Firefighter Fatality Fire. Technical Report USFA-TR-134, United States Fire Administration, December 1999. 25
- Henrik Andreasson, Tom Duckett, and Achim Lilienthal. Mini-SLAM: Minimalistic Visual SLAM in Large-Scale Environments Based on a New Interpretation of Image Similarity. In *Proc. of ICRA*, Rome, Italy, 2007. 126
- Michael Angermann, Patrick Robertson, Thomas Kemptner, and Mohammed Khider. A High Precision Reference Data Set for Pedestrian Navigation using Foot-Mounted Inertial Sensors. In *Proc. of IPIN*, Zurich, Switzerland, 2010. IEEE. 111, 114

- M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2 2002. 82
- Aline Baggio and Koen Langendoen. Monte-Carlo Localization for Mobile Wireless Sensor Networks. *Ad Hoc Networks*, 6:718–733, 2008. 11, 54
- Tim Bailey and Hugh Durrant-Whyte. Simultaneous Localisation and Mapping (SLAM): Part II State of the Art. *IEEE Robotics and Automation Magazine*, 13(3):108–117, 2006. 15, 125, 126, 129
- Tim Bailey, Juan Nieto, and Eduardo Nebot. Consistency of the FastSLAM Algorithm. In *IEEE International Conference on Robotics and Automation*, Orlando, Florida, USA, 2006. 15, 127
- Manuel Bandala and Malcolm Joyce. Wireless inertial sensor for tumour motion tracking. *Journal of Physics: Conference Series*, 76, 2007. doi: 10.1088/1742-6596/76/1/012036. 2
- Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded Up Robust Features. *Computer Vision and Image Understanding*, 110(3):346–359, 2008. 22
- Stéphane Beauregard, Widyawan, and Martin Klepal. Indoor PDR Performance Enhancement using Minimal Map Information and Particle Filters. In *Proc. of IEEE/ION PLANS*, Monterey, CA, USA, 2008. 19
- Ozkan Bebek, Michael A. Suster, Srihari Rajgopal, Michael J. Fu, Xuemei Huang, M. Cenk Çavuşoğlu, Darrin J. Young, Mehran Mehregany, Antonie J. van den Bogert, and Carlos H. Mastrangelo. Personal Navigation via High-Resolution Gait-Corrected Inertial Measurement Units. *IEEE Transactions on Instrumentation and Measurement*, 59(11):3018–3027, 11 2010. 100, 112, 117
- Fabio Belloni, Ville Ranki, Antti Kainulainen, and Andreas Richter. Angle-based Indoor Positioning System for Open Indoor Environments. In *Proc. of WPNC*, pages 261–265, Hannover, Germany, 2009. IEEE. 1, 12
- Ross Bencina and Martin Kaltenbrunner. The Design and Evolution of Fiducials for the reacTIVision System. In *Proceedings of the 3rd International Conference on Generative Systems in the Electronic Arts*, Melbourne, Australia, 2005. 64
- Charles Bibby and Ian Reid. Simultaneous Localisation and Mapping in Dynamic Environments (SLAMIDE) with Reversible Data Association. In *Proc.*

- of Robotics: Science and Systems Conference*, Philadelphia, Pennsylvania, USA, 2007. 23
- I. Bilik and J. Tabrikian. Maneuvering Target Tracking Using the Nonlinear Non-Gaussian Kalman Filter. In *Proc. of International Conference on Acoustics, Speech and Signal Processing*, Toulouse, France, 2006. 21
- Urs Bischoff, Martin Strohbach, Mike Hazas, and Gerd Kortuem. Constraint-Based Distance Estimation in Ad-Hoc Wireless Sensor Networks. In Kay Römer, Holger Karl, and Friedemann Mattern, editors, *Proc. of EWSN*, volume 3868 of *Lecture Notes in Computer Science*, pages 54–68. Springer, 2006. ISBN 3-540-32158-6. doi: 10.1007/11669463_7. 80
- Jose-Luis Blanco, Juan-Antonio Fernández-Madrigal, and Javier González. Efficient Probabilistic Range-Only SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008a. 129
- Jose Luis Blanco, Javier González, and Juan-Antonio Fernández-Madrigal. Subjective local maps for hybrid metric-topological SLAM. *Robotics and Autonomous Systems*, 57:64–74, 2008b. 23
- Alex Brooks and Tim Bailey. HybridSLAM: Combining FastSLAM and EKF-SLAM for Reliable Mapping. In *Algorithmic Foundation of Robotics VIII*, volume 57 of *Springer Tracts in Advanced Robotics*, pages 647–661. Springer Berlin/Heidelberg, 2009. 127
- Stefano Burigat and Luca Chittaro. Navigation in 3D virtual environments: Effects of user experience and location-pointing navigation aids. *International Journal of Human-Computer Studies*, 65:945–958, 2007. 202
- Jonas Callmer, David Törnqvist, and Fredrik Gustafsson. Probabilistic Stand Still Detection using Foot Mounted IMU. In *Proc. 13th International Conference on Information Fusion*, Edinburgh, Scotland, 2010. 100
- S. Challa, M. Palaniswami, and A. Shilton. Distributed Data Fusion Using Support Vector Machines. In *Proc. of ISIF*, Annapolis, Maryland, USA, 2002. 82
- Ying Chen, Zhen Cheng, and Shuliang Wen. Nonlinear Filters for Tracking Maneuverable Ballistic Missile Targets on Reentry. In *IET International Radar Conference*, Guillin, China, 2009. 9
- Krishna Kant Chintalapudi, Amit Dhariwal, Ramesh Govindan, and Gaurav Sukhatme. Ad-hoc localization using ranging and sectoring. In *INFOCOM*

2004. *Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 4, pages 2662–2672, 2004. 53
- Burcu Cinaz and Holger Kenn. HeadSLAM - Head-mounted Simultaneous Localization and Mapping for Wearable Computing Applications. In *Adjunct Proceedings of Pervasive 2008*, pages 9–13, Sydney, Australia, 2008a. Austrian Computer Society. 36
- Burcu Cinaz and Holger Kenn. HeadSLAM - simultaneous localization and mapping with head-mounted inertial and laser range sensors. In *12th IEEE International Symposium on Wearable Computers*, pages 3–10. IEEE Computer Society, 2008b. 130
- William E. Clark. *Firefighting principles & practices*. PennWell Corporation, Tulsa, OK, 1991. 28
- Laura A. Clemente, Andrew J. Davison, Ian D. Reid, José Neira, and Juan D. Tardós. Mapping Large Loops with a Single Hand-Held Camera. In *Proc. of Robotics: Science and Systems*, Atlanta, GA, 2007. MIT Press. 22, 129, 130
- Antonio Coronato and Massimo Esposito. Towards an implementation of Smart Hospital: a localization system for mobile users and devices. In *Sixth Annual IEEE International Conference on Pervasive Computing and Communications*, 2008. 18
- Waltenegus Dargie and Christian Poellabauer. *Fundamentals of Wireless Sensor Networks: Theory and Practice*. Wiley-Blackwell, 2010. 10
- Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 29(6):1052–1067, 2007. 22, 129
- Andrew John Davison. *Mobile Robot Navigation Using Active Vision*. PhD thesis, University of Oxford, 1998. 22, 129
- Matthew Deans and Martial Hebert. Experimental Comparison of Techniques for Localization and Mapping Using a Bearing-Only Sensor. In *Proc. of ISER*. Springer-Verlag, 2000. 129, 142
- Christian Decker, Albert Krohn, Michael Beigl, and Tobias Zimmer. The particle computer system. In *Proc. of IPSN*, pages 443–448. IEEE, 2005. 42
- M. W. M. Gaminí Dissanayake, Paul Newman, Steven Clark, Hugh F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map

- building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17 (3):229–241, 2001. 125
- Joseph Djugash, Sanjiv Singh, and Peter Corke. Further Results with Localization and Mapping using Range from Radio. In *Proceedings of International Conference on Field & Service Robotics (FSR)*, pages 231–242, 2005. 125
- Joseph Djugash, Sanjiv Singh, George Kantor, and Wei Zhang. Range-Only SLAM for Robots Operating Cooperatively with Sensor Networks. In *International Conference on Robotics and Automation*, pages 2078–2084. IEEE, 2006. 33, 125, 126, 128, 142
- Michelle K. Donnelly, William D. Davis, James R. Lawson, and Michael J. Selepak. Thermal Environment for Electronic Equipment Used by First Responders. Technical Report 1474, NIST, 2006. URL <http://www.fire.nist.gov/bfrlpubs/fire06/art001.html>. Accessed 21.09.2012. 27, 29, 31
- Eric Dorveaux. *Magneto-Inertial Navigation Principles and application to an indoor pedometer*. PhD thesis, École Nationale Supérieure des Mines de Paris, 2011. 122
- Paul Duff, Michael McCarthy, Angus Clark, Henk Muller, Cliff Randell, Shahram Izadi, Andy Boucher, Andy Law, Sarah Pennington, and Richard Swinford. A New Method for Auto-Calibrated Object Tracking. In *Proc. of Ubicomp*, Tokyo, Japan, 2005. 1, 12
- Hugh Durrant-Whyte and Tim Bailey. Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms. *IEEE Robotics and Automation Magazine*, 13(2):99–108, 2006. 15, 125, 126, 129
- Alon Efrat, Cesim Erten, David Forrester, Anand Iyer, and Stephen G. Kobourov. Force-Directed Approaches to Sensor Localization. In *8th Workshop on Algorithm Engineering and Experiments*, 2006. 11
- Eiman Elnahrawy, Xiaoyan Li, and Richard P. Martin. The Limits of Localization Using Signal Strength: A Comparative Study. In *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, 2004. doi: 10.1109/SAHCN.2004.1381942. 17
- John Elwell. Inertial navigation for the urban warrior. In *Proc. of Digitization of the Battlespace IV*, volume 3709, pages 196–204. SPIE, 1999. doi: 10.1117/12.351609. 99

- Frédéric Evennou and François Marx. Advanced Integration of WiFi and Inertial Navigation Systems for Indoor Mobile Positioning. *EURASIP Journal on Applied Signal Processing*, 2006:1–11, 2006. 19
- Rita F. Fahy. U.S. Fire Service Fatalities in Structure Fires, 1977-2000. Technical report, NFPA, 2002. 24
- W. Todd Faulkner, Robert Alwood, David W. A. Taylor, and Jane Bohlin. Altitude accuracy while tracking pedestrians using a boot-mounted IMU. In *Position Location and Navigation Symposium (PLANS)*, pages 90–96. IEEE, 2010. 118
- Raúl Feliz, Eduardo Zalama, and Jaime Gómez García-Bermejo. Pedestrian tracking using inertial sensors. *Journal of Physical Agents*, 3(1):35–43, 2009. 94
- Carl Fischer, Hans Gellersen, Dominique Guinard, Matt Oppenheim, and Sara Streng. RelateGateways: Using Spatial Context to Identify and Interact with Pervasive Services. In *Adjunct Proc. of Ubicomp*, pages 16–19, Innsbruck, Austria, 2007a. Poster and demo. 47
- Carl Fischer, Hans Gellersen, Dominique Guinard, and Matthew Oppenheim. RelateGateways: Using Spatial Context to Identify and Interact with Pervasive Services. In *Proc. of EuroSSC*, Lake District, UK, 2007b. Poster and demo. 47
- Carl Fischer, Kavitha Muthukrishnan, Mike Hazas, and Hans Gellersen. Ultrasound-Aided Pedestrian Dead Reckoning for Indoor Navigation. In *International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, pages 31–36, San Francisco, CA, USA, 2008. ACM. 34, 202
- Carl Fischer, Poorna Talkad Sukumar, and Mike Hazas. Tutorial: implementation of a pedestrian tracker using foot-mounted inertial sensors. *IEEE Pervasive Computing*, 2013. Accepted for publication. 4
- John Folkesson and Henrik Christensen. SIFT Based Graphical SLAM on a Packbot. In Christian Laugier and Roland Siegwart, editors, *Field and Service Robotics*, volume 42 of *Springer Tracts in Advanced Robotics*, pages 317–328. Springer Berlin / Heidelberg, 2008. 126
- Eric Foxlin. Pedestrian Tracking with Shoe-Mounted Inertial Sensors. *Computer Graphics and Applications, IEEE*, 25(6):38–46, 2005. 19, 94, 99, 101, 117
- A. Galstyan, B. Krishnamachari, K. Lerman, and S. Patten. Distributed online localization in sensor networks using a moving target. *Proceedings of the third international symposium on Information processing in sensor networks*, pages 61–70, 2004. 54

- Hans Gellersen, Carl Fischer, Dominique Guinard, Roswitha Gostner, Gerd Kortuem, Christian Kray, Enrico Rukzio, and Sara Streng. Supporting Device Discovery and Spontaneous Interaction with Spatial References. *Personal and Ubiquitous Computing*, 13(4):255–264, 2008. 42, 47
- Audrey Giremus and Jean-Yves Tournet. Controlling Particle Filter Regularization for GPS/INS Hybridization. In *14th European Signal Processing Conference (EUSIPCO'06)*, Florence, Italy, 2006. 1
- S. Godha and G. Lachapelle. Foot mounted inertial system for pedestrian navigation. *Measurement Science and Technology*, 7(7):75202–75211, July 2008. 19
- David K. Goldenberg, Pascal Bihler, Ming Cao, Jia Fang, Brian D. O. Anderson, A. Stephen Morse, and Y. Richard Yang. Localization in sparse networks using sweeps. In *Proc. of MobiCom*, pages 110–121, Los Angeles, CA, USA, 2006. ACM. doi: 10.1145/1161089.1161103. 13
- Matteo Golfarelli, Dario Maio, and Stefano Rizzi. Elastic Correction of Dead-Reckoning Errors in Map Building. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1998. 128, 209
- Pragun Goyal, Vinay J. Ribeiro, Huzur Saran, and Anshul Kumar. Strap-Down Pedestrian Dead-Reckoning System. In *Proc. of IPIN*, Guimarães, Portugal, 2011. 92
- Duncan Graham-Rowe. Indoor ‘sat-nav’ could save firefighters. *New Scientist*, 196 (2634):24, December 2007. 32
- Paul D. Groves. Optimising the Transfer Alignment of Weapon INS. *The Journal of Navigation*, 56:323–335, 2003. 9
- Paul D. Groves. *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*. Artech House, 2008. 56, 79, 94, 120
- Dominique Guinard. RelateGateways: An Architecture to Enable Spontaneous Mobile Spatial Interaction with Pervasive Services. Master’s thesis, Lancaster University, UK, and University of Fribourg, Switzerland, 6 2007. 9
- Dominique Guinard, Sara Streng, and Hans Gellersen. RelateGateways: A User Interface for Spontaneous Interaction with Pervasive Services. In *Mobile Spatial Interaction Workshop at ACM International Conference on Human Factors in Computing Systems, CHI 2007 - Reach Beyond*, 2007. 47

- Jose Guivant and Eduardo Nebot. Optimization of the simultaneous localization and map building algorithm for real time implementation. *IEEE Transactions on Robotics and Automation*, 17(3):242–257, 6 2001. 127
- Jose Guivant, Eduardo Nebot, and Stephan Baiker. Autonomous navigation and map building using laser range sensors in outdoor applications. *Journal of Robotic Systems*, 17:3817–3822, 2000. 125
- Yukang Guo and Mike Hazas. Localising Speech, Footsteps and Other Sounds using Resource-Constrained Devices. In *Proc. of IPSN*, Chicago, IL, USA, 2011. ACM. 11, 12
- Dominik Gusenbauer, Carsten Isert, and Jens Krösche. Self-Contained Indoor Positioning on Off-The-Shelf Mobile Devices. In *Proc. of IPIN*, Zürich, Switzerland, 2010. 20, 93
- Per-Olof Gutman and Mordekhai Velger. Tracking Targets With Unknown Process Noise Variance Using Adaptive Kalman Filtering. In *Proceedings of the 27th Conference on Decision and Control*, 1988. 21
- Jens-Steffen Gutmann and Kurt Konolige. Incremental mapping of large cyclic environments. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Daejeon, South Korea, 1999. 169
- Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions of Systems Science and Cybernetics*, 4:100–107, 1968. 187
- Andy Harter, Andy Hopper, Pete Steggles, Andy Ward, and Paul Webster. The Anatomy of a Context-Aware Application. *Wireless Networks*, 8:187–197, 2002. 41
- Mike Hazas, Christian Kray, Hans Gellersen, Henoc Agbota, Gerd Kortuem, and Albert Krohn. A relative positioning system for co-located mobile devices. In *Proc. of MobiSys*, pages 177–190, New York, NY, USA, 2005. ACM Press. ISBN 1-931971-31-5. doi: 10.1145/1067170.1067190. 11, 12, 42, 46
- Tian He, Sudha Krishnamurthy, John A. Stankovic, Tarek Abdelzaher, Liqian Luo, Radu Stoleru, Ting Yan, Lin Gu, Jonathan Hui, and Bruce Krogh. Energy-efficient surveillance system using wireless sensor networks. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 270–283, Boston, MA, USA, 2004. 2, 10

- Wilko Heuten, Niels Henze, Susanne Boll, and Martin Pielot. Tactile wayfinder: a non-visual support system for wayfinding. In *Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges*, volume 358 of *ACM International Conference Proceeding Series*. ACM New York, NY, USA, 2008. 175
- Dirk Hähnel, Wolfram Burgard, Dieter Fox, and Sebastian Thrun. An Efficient FastSLAM Algorithm For Generating Maps of Large-Scale Cyclic Environments from Raw Laser Range Measurements. In *International Conference on Intelligent Robots and Systems, 2003 (IROS 2003)*, pages 206–211. IEEE, 2003. 14, 36, 126, 129
- Dirk Hähnel, Wolfram Burgard, Dieter Fox, Ken Fishkin, and Matthai Philipose. Mapping and localization with RFID technology. In *IEEE International Conference on Robotics and Automation (ICRA'04)*, volume 1, pages 1015–1020, 2004. doi: 10.1109/ROBOT.2004.1307283. 126
- Jeffrey Hightower and Gaetano Borriello. Location Systems for Ubiquitous Computing. *IEEE Computer*, 34(8):57–66, 2001. 24
- Jeonghwan Hwang, Changsun Shin, and Hyun Yoe. Study on an Agricultural Environment Monitoring Server System using Wireless Sensor Networks. *Sensors*, 10:11189–11211, 2010. 2, 10
- International Association of Fire Chiefs. *Fundamentals of Fire Fighter Skills*. Jones and Bartlett Publishers, Inc., Burlington, MA, 2004. 28
- Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. KinectFusion: Realtime 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proc. of UIST*, Santa Barbara, CA, USA, 2011. 23
- Christophe Jacquet, Yacine Bellik, and Yolaine Bourda. A Context-Aware Locomotion Assistance Device for the Blind. In Klaus Miesenberger, Joachim Klaus, Wolfgang Zagler, and Dominique Burger, editors, *Computers Helping People with Special Needs*, volume 3118 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2004. ISBN 978-3-540-22334-4. doi: 10.1007/978-3-540-27817-7_64. 1
- Jasper Jahn, Ulrich Batzer, Jochen Seitz, Lucila Patino-Studencka, and Javier Gutiérrez Boronat. Comparison and Evaluation of Acceleration

- Based Step Length Estimators for Handheld Devices. In *Proc. of IPIN*, Zürich, Switzerland, 2010. 92
- Andrew H. Jazwinski. *Stochastic processes and filtering theory*. New York, Academic Press, 1970. 61
- Xiaodong Jiang, Jason I. Hong, Leila A. Takayama, and James A. Landay. Ubiquitous computing for firefighters: field studies and prototypes of large displays for incident command. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 679–686, New York, NY, USA, 2004. ACM Press. doi: 10.1145/985692.985778. 28
- Antonio R. Jiménez, Fernando Seco, Francisco Zampella, José C. Prieto, and Jorge Guevara. PDR with a Foot-Mounted IMU and Ramp Detection. *Sensors*, 11: 9393–9410, 2011. 20
- A.R. Jiménez, F. Seco, J.C. Prieto, and J. Guevara. Indoor Pedestrian Navigation using an INS/EKF framework for Yaw Drift Reduction and a Foot-mounted IMU. In *Proc. of WPNC*, Dresden, Germany, 2010. 94, 108
- Yunye Jin, Hong-Song Toh, Wee-Seng Soh, and Wai-Choong Wong. A Robust Dead-Reckoning Pedestrian Tracking System with Low Cost Sensors. In *IEEE International Conference on Pervasive Computing and Communications (Per-Com)*, Seattle, WA, USA, 2011. 93, 122
- Simon J. Julier and Jeffrey K. Uhlmann. Using covariance intersection for SLAM. *Robotics and Autonomous Systems*, 55:3–20, 2006. 209
- George Kantor, Sanjiv Singh, Ronald Peterson, Daniela Rus, Aveek Das, Vijay Kumar, Guilherme Pereira, and John Spletzer. Distributed Search and Rescue with Robot and Sensor Teams. In *Proceedings of the 4th International Conference on Field and Service Robotics*, volume 24. Springer Berlin / Heidelberg, 2003. 33
- Jonghyuk Kim and Salah Sukkarieh. Real-time implementation of airborne inertial-SLAM. *Robotics and Autonomous Systems (Elsevier)*, 55:62–71, 2007. 13, 125
- Markus Klann. Playing with fire: participatory design of wearable computing for fire fighters. In *CHI’07 extended abstracts on Human factors in computing systems*, pages 1665–1668, 2007. 176
- Markus Klann. Tactical Navigation Support for Firefighters: The LifeNet Ad-Hoc Sensor-Network and Wearable System. In J. Löffler and M. Klann, editors, *Mobile Response*, number 5424 in LNCS, pages 41–56. Springer Berlin / Heidelberg, 2009. 26, 27, 32, 33

- Markus Klann, Till Riedel, Hans Gellersen, Carl Fischer, Matt Oppenheim, Paul Lukowicz, Gerald Pirkel, Kai Kunze, Monty Beuster, Michael Beigl, Otto Visser, and Mirco Gerling. LifeNet: an Ad-hoc Sensor Network and Wearable System to Provide Firefighters with Navigation Support. In *Adjunct Proc. of Ubicomp*, pages 124–127, Innsbruck, Austria, 2007. Poster and demo. 48
- Lindsay Kleeman. Advanced sonar and odometry error modeling for simultaneous localisation and map building. In *Proc. of IROS*, pages 699–704, Las Vegas, NV, USA, 2003. 14
- Alexander Kleiner and Christian Dornhege. Real-time localization and elevation mapping within urban search and rescue scenarios: Field Reports. *Journal of Field Robotics*, 24(8-9):723–745, 2007. doi: 10.1002/rob.v24:8/9. 15
- Alexander Kleiner and Dali Sun. Decentralized SLAM for Pedestrians without direct Communication. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots & Systems (IROS)*, San Diego, CA, USA, 2007. 128, 130, 170
- Kamil Kloch, Carl Fischer, and Paul Lukowicz. Collaborative PDR Localisation with Mobile Phones. In *Proc. of ISWC*, San Francisco, CA, USA, 2011. IEEE. 122
- Kurt Konolige. Large-scale map-making. In *Proceedings of the National Conference on AI*, San Jose, CA, USA, 2004. 14
- Gerd Kortuem, Christian Kray, and Hans Gellersen. Sensing and visualizing spatial relations of mobile devices. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 93–102, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-271-2. doi: 10.1145/1095034.1095049. 45
- Bernhard Krach and Patrick Robertson. Cascaded Estimation Architecture for Integration of Foot-Mounted Inertial Sensors. In *IEEE/ION Position Location and Navigation Symposium 2008*, Monterey, CA, USA, 2008. 19
- Albert Krohn, Michael Beigl, Mike Hazas, Hans-Werner Gellersen, and Albrecht Schmidt. Using Fine-Grained Infrared Positioning to Support the Surface-Based Activities of Mobile Users. In *25th IEEE International Conference on Distributed Computing Systems Workshops*, pages 463–468, 2005. doi: 10.1109/ICDCSW.2005.139. 11, 42, 45
- Albert Krohn, Mike Hazas, and Michael Beigl. Removing Systematic Error in Node Localisation Using Scalable Data Fusion. In *Proc. of EWSN*, pages 341–356, Delft, The Netherlands, 2007. 13

- Branislav Kusy, Akos Ledeczi, and Xenofon Koutsoukos. Tracking Mobile Nodes Using RF Doppler Shifts. In *Proc. of SenSys*, Sydney, Australia, 2007. ACM. 55
- Koen Langendoen and Niels Reijers. Distributed localization in wireless sensor networks: a quantitative comparison. *Computer Networks*, 43:499–518, 2003. 53
- Paul U. Lee and Barbara Tversky. Interplay between Visual and Spatial: The Effect of Landmark Descriptions on Comprehension of Route/Survey Spatial Descriptions. *Spatial Cognition and Computation*, 5(2 & 3):163–185, 2005. 22
- John J. Leonard and Hans Jacob S. Feder. A Computationally Efficient Method for Large-Scale Concurrent Mapping and Localization. In *Proc. Ninth International Symposium on Robotics Research*, Snowbird, UT, 2000. Springer-Verlag. 127
- X. Rong Li and Vesselin P. Jilkov. Survey of Maneuvering Target Tracking. Part I: Dynamic Models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1333–1364, 10 2003. 21
- H. Q. Liu, H. C. So, F. K. W. Chan, and K. W. K. Lui. Distributed Particle Filter For Target Tracking In Sensor Networks. *Progress In Electromagnetics Research*, 11:171–182, 2009. 82
- David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 22
- Abdullah Al Mahmud, Omar Mubin, and Suleman Shahid. User experience with in-car GPS navigation systems: comparing the young and elderly drivers. In *Proc. of MobileHCI*, Bonn, Germany, 2009. ACM. doi: 10.1145/1613858.1613962. 1
- Daniel J. Mennill, Stephanie M. Doucet, Kara-Anne A. Ward, Dugan F. Maynard, Brian Otis, and John M. Burt. A novel digital telemetry system for tracking wild animals: a field test for studying mate choice in a lekking tropical bird. *Methods in Ecology and Evolution*, 3(4):663–672, 2012. doi: 10.1111/j.2041-210X.2012.00206.x. 2
- Leonard E. Miller. Indoor Navigation for First Responders: A Feasibility Study. Technical report, National Institute of Standards and Technology, 2006. URL http://www.antd.nist.gov/wctg/RFID/Report_indoornav_060210.pdf. Accessed 2012.09.21. 35
- Esmond Mok and Günther Retscher. Location Determination Using WiFi – Fingerprinting Versus WiFi – Trilateration. *Journal of Location Based Services*, 1(2):145–159, 2007. 17

- Michael Montemerlo. *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem With Unknown Data Association*. PhD thesis, Carnegie Mellon University, 2003. 127
- Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI'02)*, pages 593–598, Edmonton, Alberta, Canada, 2002. The AAAI Press. 127
- J.M.M. Montiel, Javier Civera, and Andrew J. Davison. Unified Inverse Depth Parametrization for Monocular SLAM. In *Proceedings of Robotics: Science and Systems*, Philadelphia, PA, USA, 2006. 129
- David Moore, John Leonard, Daniela Rus, and Seth Teller. Robust distributed network localization with noisy range measurements. In *Proc. of SenSys*, pages 50–61, Baltimore, Maryland, USA, 2004. doi: 10.1145/1031495.1031502. 53, 72
- Bojan Mrazovac, Milan Z. Bjelica, Istvan Papp, and Nikola Teslic. Smart Audio/Video Playback Control Based on Presence Detection and User Localization in Home Environment. In *Second Eastern European Regional Conference on the Engineering of Computer Based Systems*, 2011. 18
- Kavitha Muthukrishnan and Mike Hazas. Position Estimation from UWB Pseudorange and Angle-of-Arrival: A Comparison of Non-linear Regression and Kalman Filtering. In *Location and Context Awareness*, Tokyo, Japan, 2009. Springer-Verlag. 12, 147
- National Fire Protection Association. NFPA 1500 – Standard on Fire Department Occupational Safety and Health Program. Technical report, National Fire Protection Association, Quincy, MA, 2002. 27
- Richard A. Newcombe and Andrew J. Davison. Live Dense Reconstruction with a Single Moving Camera. In *Proceedings of CVPR*, San Francisco, CA, USA, 2010. 22, 129
- P. Newman and J. Leonard. Pure Range-Only Sub-Sea SLAM. In *Proceedings of the IEEE Conference on Robotics and Automation (ICRA)*, pages 1921–1926, Taipei, Taiwan, 2003. 125, 129
- Dragos Niculescu and Badri Nath. DV Based Positioning in Ad Hoc Networks. *Telecommunication Systems*, 22(1-4):267–280, 2003. 53

- Ewa Niewiadomska-Szynkiewicz, Piotr Kwaśniewski, and Izabela Windyga. Comparative Study of Wireless Sensor Networks Energy-Efficient Topologies and Power Save Protocols. *Journal of Telecommunications and Information Technology*, 3:68–75, 2009. 1
- John-Olof Nilsson, Isaac Skog, and Peter Händel. Performance characterisation of foot-mounted ZUPT-aided INSs and other related systems. In *Proc. of IPIN*, pages 182–183, Zurich, Switzerland, 2010. Extended abstract. 119
- Mary Catherine O'Connor. Caterpillar Looks to RFID to Improve Work-in-Progress Visibility. RFID Journal (online), February 2007. URL <http://www.rfidjournal.com/article/view/3038/>. Accessed 2012.09.21. 16
- Lauro Ojeda and Johann Borenstein. Non-GPS Navigation for Security Personnel and First Responders. *Journal of Navigation*, 60(3):391–407, 9 2007. 94
- Edwin Olson, John Leonard, and Seth Teller. Robust range-only beacon localization. *IEEE Journal of Oceanic Engineering*, 31(4):949–958, 10 2006. 13, 125, 129
- Amitangshu Pal. Localization Algorithms in Wireless Sensor Networks: Current Approaches and Future Challenges. *Network Protocols and Algorithms*, 2(1):45–74, 2010. 12
- Jun-geun Park, Erik D. Demaine, and Seth Teller. Moving-Baseline Localization. In *International Conference on Information Processing in Sensor Networks*, St Louis, Missouri, USA, 2008. 4, 53
- Sang Kyeong Park and Young Soo Suh. A Zero Velocity Detection Algorithm Using Inertial Sensors for Pedestrian Navigation Systems. *Sensors*, 10:9163–9178, 2010. 99
- Rong Peng and Mihail L. Sichitiu. Angle of Arrival Localization for Wireless Sensor Networks. In *Proc. of SECON*, pages 374–382, Reston, VA, USA, 2006. 12
- Nissanka B. Priyantha, Allen K. L. Miu, Hari Balakrishnan, and Seth Teller. The Cricket Compass for ContextAware Mobile Applications. In *7th Annual International Conference on Mobile Computing and Networking*, pages 1–14. ACM New York, NY, USA, 2001. 12
- Nissanka B. Priyantha, Hari Balakrishnan, Erik Demaine, and Seth Teller. Anchor-Free Distributed Localization in Sensor Networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys'03)*, pages 340–341, 2003. 11

- Honghui Qi and John B. Moore. Direct Kalman filtering approach for GPS/INS integration. *IEEE Transactions on Aerospace and Electronic Systems*, 38(2): 687–693, 4 2002. 109, 110
- Cliff Randell, Chris Djiallis, and Henk Muller. Personal Position Measurement Using Dead Reckoning. In *Proc. of ISWC*, pages 166–176, White Plains, NY, USA, 2003. IEEE Computer Society. 18
- Valérie Renaudin, Okan Yalak, Phillip Tomé, and Bertrand Merminod. Indoor Navigation of Emergency Agents. *European Journal of Navigation*, 5(3):36–45, July 2007. 32, 35
- Injong Rhee, Ajit Warrier, Mahesh Aia, Jeongki Min, and Mihail Sichitiu. Z-MAC: a Hybrid MAC for Wireless Sensor Networks. *IEEE/ACM Transactions on Networking*, 16(3):511–524, 6 2008. 81
- Bernhard E. Riecke, Joerg Schulte-Pelkum, and Heinrich H. Buelthoff. Perceiving simulated ego-motions in virtual reality: comparing large screen displays with HMDs. In *Proc. SPIE 5666, Human Vision and Electronic Imaging X*, San Jose, CA, USA, 2005. doi: 10.1117/12.610846. 198
- Patrick Robertson, Michael Angermann, and Bernhard Krach. Simultaneous Localization and Mapping for Pedestrians using only Foot-Mounted Inertial Sensors. In *Proceedings of UbiComp 2009*, Orlando, FL, 2009. ACM. 122, 131
- Patrick Robertson, Michael Angermann, and Mohammed Khider. Improving Simultaneous Localization and Mapping for pedestrian navigation and automatic mapping of buildings by using online human-based feature labeling. In *Proc. of PLANS*, Indian Wells, CA, USA, 2010. IEEE. 122
- Enrico Rukzio, Michael Mueller, and Robert Hardy. Design, Implementation and Evaluation of a Novel Public Display for Pedestrian Navigation: The Rotating Compass. In *Proc. of CHI*, Boston, MA, USA, 2009. 175
- Andreas Savvides, Heemin Park, and Mani B. Srivastava. The Bits and Flops of the N-hop Multilateration Primitive For Node Localization Problems. In *Proc. of WSNA*, Atlanta, Georgia, USA, 2002. ACM Press. ISBN 1-58113-589-0. 52, 55
- Timothy E. Sendelbach. Search Line Survival Training. Technical report, TES2 Training, Savannah, GA, 2002. URL http://www.tes2training.com/handouts/search_line_survival_training.pdf. Accessed 2012.09.21. 27

- Y. Shang and W. Ruml. Improved MDS-based localization. 4:2640–2651, 2004. 53, 60, 71, 73, 128
- Larry Sher. Comment on GPS Forums. Website., 2003. URL <http://www.gps-forums.net/re-question-regarding-inertial-navigation-systems-t25879.html>. Accessed 2012.09.21. 99
- Jianbo Shi and Carlo Tomasi. Good Features to Track. In *Proc. of Computer Vision and Pattern Recognition*, pages 593–600, Seattle, WA, 1994. IEEE. 22
- Frank Siegemund and Christian Flöer. Interaction in Pervasive Computing Settings Using Bluetooth-Enabled Active Tags and Passive RFID Technology Together with Mobile Phones. In *Proc. of PerCom*, pages 378–388, Dallas-Fort Worth, TX, USA, 2003. IEEE Computer Society. 9
- Dan Simon. From Here to Infinity. *Embedded Systems Programming*, July:20–32, 2000. 9
- Dan Simon. Kalman Filtering. *Embedded Systems Programming*, June:72–79, 2001. 93
- Dan Simon. *Optimal State Estimation*. John Wiley & Sons, Inc., Hoboken, NJ, first edition, 2006. 56, 61, 62, 82, 110, 140
- Gyula Simon, Miklós Maróti, Ákos Lédeczi, György Balogh, Branislav Kusy, András Nádas, Gábor Pap, János Sallai, and Ken Frampton. Sensor network-based countersniper system. In *Proc. of SenSys*, pages 1–12, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-879-2. doi: 10.1145/1031495.1031497. 9
- Robert A. Singer. Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets. *IEEE Transactions on Aerospace and Electronic Systems*, 6(4):473–483, 7 1970. 21
- Isaac Skog, Peter Händel, Jouni Rantakokko, and John-Olof Nilsson. Zero velocity detection — An algorithm evaluation. *IEEE Transactions on Biomedical Engineering*, 57(11):2657–2666, 2010a. 108, 112
- Isaac Skog, John-Olof Nilsson, and Peter Händel. Evaluation of Zero-Velocity Detectors for Foot-Mounted Inertial Navigation Systems. In *2010 International Conference on Indoor Positioning and Indoor Navigation*, pages 172–173, 2010b. Extended abstract. 100

- Randall C. Smith and Peter Cheeseman. On the Representation and Estimation of Spatial Uncertainty. *The International Journal of Robotics Research*, 5(4): 56–68, 1986. 126
- Taek Lyul Song. Observability of Target Tracking with Range-Only Measurements. *IEEE Journal of Oceanic Engineering*, 24:383–387, 1999. 157
- Olivier Stasse, Andrew J. Davison, Ramzi Sellaouti, and Kazuhito Yokoi. Real-time 3D SLAM for Humanoid Robot considering Pattern Generator Information. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, page 348–355, Beijing, China, 2006. 13
- Igor Steiner, Clemens Bürgib, Stefan Werffelib, Giacomo Dell’Omo, Paolo Valenti, Gerhard Tröster, David P. Wolfer, and Hans-Peter Lipp. A GPS logger and software for analysis of homing in pigeons and small mammals. *Physiology & Behavior*, 71(5):589–596, 12 2000. doi: 10.1016/S0031-9384(00)00409-1. 2
- Daniel Steingart, Joel Wilson, Andrew Redfern, Paul Wright, Russell Romero, and Lloyd Lim. Augmented Cognition for Fire Emergency Response: An Iterative User Study. In *Augmented Cognition, Proceedings of the 11th International Conference on Human-Computer Interaction (HCI)*, Las Vegas, NV, 2005. 29, 32
- M. Stojanovic, L. Freitag, J. Leonard, and P. Newman. A Network Protocol for Multiple AUV Localization. In *Proc. of IEEE OCEANS Conference*, Biloxi, Mississippi, USA, 2002. 2
- Radu Stoleru, Tian He, and John A. Stankovic. Range-free Localization. In Radha Poovendran, Cliff Wang, and Sumit Roy, editors, *Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks*, volume 30 of *Advances in Information Security*. Springer, 2007. 11
- Hauke Strasdat, J.M.M. Montiel, and Andrew J. Davison. Scale Drift-Aware Large Scale Monocular SLAM. In *Proc. of Robotics: Science and Systems*, Seattle, WA, USA, 2010. 130
- Peter Strömbäck, Jouni Rantakokko, Sven-Lennart Wirkander, Mikael Alexander-sson, Karina Fors, Isaac Skog, and Peter Händel. Foot-Mounted Inertial Navigation and Cooperative Sensor Fusion for Indoor Positioning. Technical report, KTH, Stockholm, Sweden, 2009. 122, 169
- Jackrit Suthakorn, Syed Saqib Hussain Shah, Suratana Jantarajit, Woratit Onprasert, Watcharawit Saensupo, Supawat Saeung, Sakol Nakdhamabhorn, Vera Sa-Ing, and Sureerat Reaungamornrat. On the Design and Development of

- A Rough Terrain Robot for Rescue Missions. In *International Conference on Robotics and Biomimetics*, Guilin, Guangxi, China, 2009. 15
- Jean-Philippe Tardif, Yanis Pavlidis, and Kostas Daniilidis. Monocular Visual Odometry in Urban Environments Using an Omnidirectional Camera. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nice, France, 2008. 14, 129
- Christopher Taylor, Ali Rahimi, Jonathan Bachrach, Howard Shrobe, and Anthony Grue. Simultaneous localization, calibration, and tracking in an ad hoc sensor network. In *Proc. of IPSN*, Nashville, TN, USA, 2006. ISBN 1-59593-334-4. 52, 54, 80
- Bjorn Thorbjornsen, Neil White, Andrew Brown, and Jeff Reeve. Radio Frequency (RF) Time-of-Flight Ranging for Wireless Sensor Networks. *Measurement Science and Technology*, 21(3):1–12, 2010. 11
- S. Thrun, M. Beetz, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. Probabilistic Algorithms and the Interactive Museum Tour-Guide Robot Minerva. *International Journal of Robotics Research*, 19(11):972–999, 2000. 1
- Sebastian Thrun. A Probabilistic On-Line Mapping Algorithm for Teams of Mobile Robots. *The International Journal of Robotics Research*, 20:335–363, 2001. 128, 170
- Sebastian Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002. 125
- Sebastian Thrun and Michael Montemerlo. The GraphSLAM Algorithm with Application to Large-Scale Mapping of Urban Structures. *The International Journal of Robotics Research*, 25(5–6):403–429, June 2006. 127, 128, 155, 170
- Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots. *Machine Learning and Autonomous Robots*, 31(5):1–25, 1998. 127
- Sebastian Thrun, Yufeng Liu, Daphne Koller, Andrew Y. Ng, Zoubin Ghahramani, and Hugh Durrant-Whyte and. Simultaneous Localization and Mapping with Sparse Extended Information Filters. *The International Journal of Robotics Research*, 23(7–8):693–716, 2004. 127

- David H. Titterton and John L. Weston. *Strapdown Inertial Navigation Technology*. Institution of Engineering and Technology, second edition, 2004. 93
- Ubisense. *Ubisense Series 7000 IP Sensors Fact Sheet Rev 5 EN120822*, 2012. URL http://www.ubisense.net/en/media/pdfs/factsheets_pdf/83188_series_7000_ip_sensors.pdf. Accessed 2012.09.21. 133
- Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, M. N. Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, Michele Gittleman, Sam Harbaugh, Martial Hebert, Thomas M. Howard, Sascha Kolski, Alonzo Kelly, Maxim Likhachev, Matt McNaughton, Nick Miller, Kevin Peterson, Brian Pilnick, Raj Rajkumar, Paul Rybski, Bryan Salesky, Young-Woo Seo, Sanjiv Singh, Jarrod Snider, Anthony Stentz, William “Red” Whittaker, Ziv Wolkowicki, Jason Ziglar, Hong Bae, Thomas Brown, Daniel Demitrish, Bakhtiar Litkouhi, Jim Nickolaou, Varsha Sadekar, Wende Zhang, Joshua Struble, Michael Taylor, Michael Darms, and Dave Ferguson. Autonomous driving in urban environments: Boss and the Urban Challenge. *Journal of Field Robotics*, 25(8): 425–466, 2008. ISSN 1556-4967. doi: 10.1002/rob.20255. 1, 13
- Marco Vala, Guilherme Raimundo, Pedro Sequeira, Pedro Cuba, Rui Prada, Carlos Martinho, and Ana Paiva. ION Framework - A Simulation Environment for Worlds with Virtual Agents. In *Proc. of IVA*, Amsterdam, The Netherlands, 2009. doi: 10.1007/978-3-642-04380-2_45. 197
- Juan Valverde, Victor Rosello, Gabriel Mujica, Jorge Portilla, Amaia Uriarte, and Teresa Riesgo. Wireless Sensor Network for Environmental Monitoring: Application in a Coffee Factory. *International Journal of Distributed Sensor Networks*, 2012. doi: 10.1155/2012/638067. 10
- Rudolph van der Merwe. *Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models*. PhD thesis, OGI School of Science & Engineering at Oregon Health & Science University, 2004. 9
- Luis von Ahn. Games with a Purpose. *IEEE Computer*, June:96–98, 2006. 198
- Kevin J. Walchko, Michael C. Nechyba, Eric Schwartz, and Antonio Arroyowal. Embedded Low Cost Inertial Navigation System. In *Florida Conference on Recent Advances in Robotics, FAU*, 2003. 14
- Ulrich Walder, Thomas Bernoulli, and Thomas Wießflecker. An Indoor Positioning System for Improved Action Force Command and Disaster Management. In *Proceedings of the 6th International ISCRAM Conference*, 2009. 32

- Bruce N. Walker and Jeffrey Lindsay. Navigation Performance With a Virtual Auditory Display: Effects of Beacon Sound, Capture Radius, and Practice. *Human Factors*, 48(2):265–278, 2006. 175
- Sheng Wan and Eric Foxlin. Improved Pedestrian Navigation Based on Drift-Reduced MEMS IMU Chip. In *Proc. ION 2010 International Technical Meeting*, pages 220–229, San Diego, CA, 2010. 122, 150
- Xiaoping Wang, Jun Luo, Shanshan Li, Dezun Dong, and Weifang Cheng. Component based localization in sparse wireless ad hoc and sensor networks. In *Proc. of ICNP*, Orlando, FL, USA, 2008. 12
- Greg Welch and Gary Bishop. SCAAT: Incremental Tracking with Incomplete Information. *Computer Graphics*, 31:333–344, 1997. 46, 50, 55
- Greg Welch and Gary Bishop. An Introduction to the Kalman Filter. Technical Report TR 95-041, University of North Carolina, Chapel Hill, 2006. URL <http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html>. Accessed 2012.09.21. 93
- WGS84. Department of Defense World Geodetic System 1984, Its Definition and Relationships With Local Geodetic Systems. Technical Report Third edition, amendment 1, National Geospatial-Intelligence Agency, 2000. 40
- Kamin Whitehouse and David Culler. *A robustness analysis of multi-hop ranging-based localization approximations*. ACM Press, Nashville, Tennessee, USA, 2006. ISBN 1-59593-334-4. 53, 54, 56, 66
- Kamin Whitehouse, Chris Karlof, and David Culler. A practical evaluation of radio signal strength for ranging-based localization. *SIGMOBILE Mob. Comput. Commun. Rev.*, 11(1):41–52, 2007. doi: 10.1145/1234822.1234829. 11
- Widyawan, Martin Klepal, and Stéphane Beauregard. A Backtracking Particle Filter for Fusing Building Plans with PDR Displacement Estimates. In *Proceedings of the 5th Workshop on Positioning, Navigation and Communication (WPNC’08)*, pages 207–212, Hannover, Germany, 2008. IEEE. 35
- Joel Wilson, Vikas Bhargava, Andrew Redfern, and Paul Wright. A Wireless Sensor Network and Incident Command Interface for Urban Firefighting. In *Fourth Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services*, pages 1–7, Philadelphia, PA, USA, 2007. IEEE. 33
- Bob G. Witmer, Wallace J. Sadowski, and Neal M. Finkelstein. VE-Based Training Strategies for Acquiring Survey Knowledge. *Presence*, 11(1):1–18, February 2002. 175, 185

- Oliver Woodman and Robert Harle. Pedestrian Localisation for Indoor Environments. In *Proceedings of the 10th international conference on Ubiquitous computing (Ubicomp)*, volume 344, pages 114–123, Seoul, South Korea, 2008. 35
- Mike Worrell and Andy MacFarlane. Phoenix Fire Department Radio System Safety Project. Technical report, City of Phoenix Fire Department, Phoenix, AZ, 2004. 29
- Feng Xia, Xue Yang, Haifeng Liu, Da Zhang, and Wenhong Zhao. Energy-Efficient Opportunistic Localization with Indoor Wireless Sensor Networks. *Computer Science and Information Systems*, 8(4):973–990, 2011. 10
- Bin Xiao, Hekang Chen, and Shuigeng Zhou. Distributed Localization Using a Moving Beacon in Wireless Sensor Networks. *Transactions on Parallel and Distributed Systems*, 15(9):587–600, 2008. 11